start.js:

```
var app = require('./server/app');

if (app.locals.sslOptions) {
    var http = require('https');
} else {
    var http = require('http');
}

var port=process.env.PORT || '3000';
app.set('port', port);

if (app.locals.sslOptions) {
    var server = http.createServer(app.locals.sslOptions, app);
} else {
    var server = http.createServer(app);
}

server.listen(port);
console.log('Listening on ' + port);
```

server/app.js:

```
var express = require('express');
var session = require('express-session');
var path = require('path');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');

var fs = require('fs');
var MongoStore = require('connect-mongo')(session);

var routes = require('./routes');
var app = express();

app.locals.sslOptions = {
    key: fs.readFileSync('keys/comp2406-test-key.pem'),
    cert: fs.readFileSync('keys/comp2406-test-cert.pem')
};

app.set('views', 'views');
app.set('view engine', 'jade');

app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(session({secret: 'superSekretHere!',
                 resave: false,
                 saveUninitialized: false,
                 store: new MongoStore(
                     {url: 'mongodb://localhost/upload-demo'})}));

app.use(express.static('public'));
app.use('/', routes);

module.exports = app;
```

server/routes.js:

```
1  var express = require('express');
2  var router = express.Router();
3
4  var bcrypt = require('bcrypt');
5  var multer  = require('multer')
6  var storage = multer.memoryStorage()
7  var upload = multer({ storage: storage })
8
9  var mc = require('mongodb').MongoClient;
10 var db, filesCollection, usersCollection;
11
12 var connectCallback = function(err, returnedDB) {
13     if (err) {
14         throw err;
15     }
16
17     db = returnedDB;
18
19     filesCollection = db.collection('files');
20     usersCollection = db.collection('users');
21 }
22
23 mc.connect('mongodb://localhost/upload-demo', connectCallback);
24
25 router.get('/', function (req, res) {
26     if (req.session.username) {
27         res.redirect("/users");
28     } else {
29         res.render('index', { title: 'COMP 2406 Final: Login',
30                               error: req.query.error });
31     }
32 });
33
34 router.get('/getFileStats', function(req, res) {
35     function returnStats(err, stats) {
36         if (err) {
37             sendStatus(500);
38         } else {
39             res.send(stats);
40         }
41     }
42
43     if (req.session.username) {
44         filesCollection.find({owner: req.session.username}, {content: 0})
45             .toArray(returnStats);
46     } else {
47         res.send("ERROR: not logged in");
48     }
49 });
50
51 router.get('/users', function (req, res) {
52     if (req.session.username) {
53         res.render("account.jade", {username:req.session.username,
54                                     title: "COMP 2406 Final: Account"});
55     } else {
56         res.redirect("/?error=Not Logged In");
57     }
58 });
```

```
59
60
61
62 router.post("/uploadText", upload.single('theFile'), function(req, res) {
63     var theFile = req.file;
64     var storedFile;
65
66     function returnResult(err, result) {
67         if (err) {
68             res.sendStatus(500);
69         } else {
70             res.send("Upload succeeded: " + storedFile.name + "\n");
71         }
72     }
73
74     if (theFile && (theFile.mimetype === 'text/plain')) {
75         storedFile = {
76             name: theFile.originalname,
77             owner: req.session.username,
78             size: theFile.size,
79             content: theFile.buffer.toString('utf8')
80         };
81         filesCollection.update({name: theFile.originalname},
82                                storedFile,
83                                {upsert: true},
84                                returnResult);
85     } else {
86         res.sendStatus(403);
87     }
88 });
89
90 router.post("/login", function(req, res) {
91     var username = req.body.username;
92     var password = req.body.password;
93     var user;
94
95     function passwordResult(err, authenticated){
96         if (authenticated) {
97             req.session.username = user.username;
98             res.redirect("/users");
99         } else {
100            res.redirect("/?error=invalid username or password");
101        }
102    }
103
104    function usernameResult(err, users){
105        if (err || users.length===0){
106            res.redirect("/?error=invalid username or password");
107            return;
108        }
109
110        user = users[0];
111        bcrypt.compare(password, user.password, passwordResult);
112    }
113
114    usersCollection.find({username: username}).toArray(usernameResult);
115 });
116
117 router.post("/logout", function(req, res) {
118     req.session.destroy(function(err) {
```

```
119         res.redirect("/");
120     });
121 });
122
123
124 router.post("/register", function(req, res) {
125     var username = req.body.username;
126     var password = req.body.password;
127
128     function reportUpdateResult(err, newUsers) {
129         var status;
130         if (err) {
131             status = "failed";
132         } else {
133             status = "succeeded";
134         }
135         res.render('registered', { title: "Registration " + status,
136                                    username: username, status: status});
137     }
138
139     function storeSaltedPassword(err, hash) {
140         var newUser = {
141             username: username,
142             password: hash,
143         };
144
145         usersCollection.update({username: username}, newUser,
146                                {upsert: true}, reportUpdateResult);
147     }
148
149     function addUser(err, users) {
150         if (users.length !== 0) {
151             res.redirect("/?error=user already exists");
152         } else {
153             bcrypt.genSalt(10, function(err, salt) {
154                 bcrypt.hash(password, salt, storeSaltedPassword)});
155         }
156     }
157
158     usersCollection.find({username: username}).toArray(addUser);
159 });
160
161 router.post("/downloadFile", function (req, res) {
162     var name = req.body.downloadFile;
163
164     function returnFile(err, theFile) {
165         if (err) {
166             res.send("File not found");
167         } else {
168             res.type('text/plain');
169             res.send(theFile.content);
170         }
171     }
172
173     if (!req.session.username) {
174         res.send("Not logged in.");
175     } else {
176         filesCollection.findOne({name: name, owner: req.session.username},
177                                 returnFile);
178     }
```

```
});

module.exports = router;
```

public/account.js:

```
$(function() {
    var fileList = $("#fileList");

    var stats;

    function downloadFile(i) {
        function saveDownloadedFile(fileContents) {
            console.log("Trying to save file");
            console.log(fileContents);
            saveAs(new Blob([fileContents],
                            {type: "text/plain;charset=utf-8"}),
                   stats[i].name);
        }

        return function() {
            $.post("/downloadFile", {downloadFile: stats[i].name},
                   saveDownloadedFile);
        }
    }

    function doUpdateFileList (returnedStats) {
        var i;
        stats = returnedStats;
        fileList.empty();
        for (i=0; i<stats.length; i++) {
            fileList.append('<li> <a id="file' + i + '" href="#">' +
                            stats[i].name +
                            "</a> (" + stats[i].size + " bytes)");
            $("#file" + i).click(downloadFile(i));
        }
    }

    function updateFileList () {
        $.getJSON("/getFileStats", doUpdateFileList);
    }

    updateFileList();

    $("#fileuploader").uploadFile({
        url:"/uploadText",
        fileName:"theFile",
        dragDrop: false,
        uploadStr: "Upload Files",
        afterUploadAll: updateFileList
    });
});
```

public/register.js:

```
$(function(){
    $("#register").on("click",function(){
        var $form = $("form");
        $form.attr("action","/register");
        $form.submit();
    });
});
```

views/layout.jade:

```
doctype html
html
  head
    title= title
    link(rel='stylesheet', href='/style.css')
    script(src='/jquery-1.12.1.js')
    block header
  body
    block content
```

views/index.jade:

```
extends layout

block header
  script(src='/register.js')

block content
  h1 COMP 2406 Final Exam Demo
  - if(error)
    div.alert-error #{error}
  p Please log in
  div
    form(action="/login", method="post")
        div.control-group.input-append
            input(type="text", name="username")
            label.add-on(for="username") Username
        div.control-group.input-append
            input(type="password", name="password")
            label.add-on(for="password") Password
        button(type="submit") Login
        button#register(type="button") Register
```

views/account.jade:

```
extends layout

block header
  link(rel='stylesheet', href="/uploadfile.css")
  script(src="/jquery.form.js")
  script(src="/jquery.uploadfile.js")
  script(src="/FileSaver.js")
  script(src="/account.js")

block content
  h1 File storage demo
  h3 User: #{username}
  form(action="/logout", method="post")
        button(type="submit") Logout
  h2 Files uploaded:

  div
    ul#fileList

  div#fileuploader
```