<< Boundary Class >>
Login Page Class

- userName: string
- userPassword: string
- passwordHash: string
- firstName: string
- lastName: string
- email: string

- getUserName() : void
- getUserPassword() : void
- getFirstName() : void
- getLastName() : void
- getEmail() : void
- encryptPassword(password: string) : string
- authenticate(userName: string, passwordHash: string) : boolean
- addNewUser(userName: string, passwordHash: string) : void

<< Entity Class >>
Registered Users Database

- userID: int
- userName: string
- passwordHash: string
- firstName: string
- lastName: string
- email: string

- getPasswordHash(username: string) : string
- addUser(username: string, password: string) : void
- updatePassword(username: string, password: string) : void

<< Boundary Class >>
Editor Class

- ebookName: string
- ebookDescription: string
- ebookContent: ebook
- ebookOwner: string
- isPublic: boolean
- fileType: string

- createNewEbook(name: string, owner: string description: string) : void
- loadEbook(name: string) : void
- updateDescription(description: string) : void
- togglePublicStatus() : void
- getEbookToPublish() : void

- getFileType() : void
- publishEbook(name: string) : void
- saveEbook(name: string, content: string) : void

<< Control Class >>
Document Converter Class

- ebookName: string
- outputFileType: string
- inputContent: string
- outputContent: string
- outputFilename: string

- loadInputContent(name: string): string
- generateOutputContent(inputContent: string, outputFileType): string
- writeOutputContent(outputContent string, outputFileName: string): void

<< Entity Class >>
eBooks Database

- ID: int
- name: string
- owner: string
- content: string
- fileType: string
- description: string
- isPublic: boolean

- updateDescription(description: string) : void
- getContent(name: string) : string
- getOwner(name: string) : string
- getDescription(name: string) : string
- writeEbook(name: string, content: string) : void
- setPublic(name: string) : void
- setPrivate(name: string) : void
- isPublic(name: string) : boolean

<< Control Class >>
Document Initializer Class

- ebookName: string
- ebookOwner: string

- getEbookName() : string
- getEbookOwner() : string
- addEbookToDatabase(name: string, owner: string) : void

<< Boundary Class >> **View Class**
• ebookName: string
• getContent(name: string) : string • display(content: string) : void

Detailed Design:

Login Page Class:

```
void getUserName()
{
      output "User name: ";
      userName = input;

void getUserPassword()
{
      output "Password: ";
      userPassword = input;
      passwordHash = encryptPassword(userPassword);

void getFirstName()
{
      output "First name: ";
      fistName = input;

void getLastName()
{
      output "Last name: ";
      lastName = input

void getEmail()
{
      output "Email: ";
      email = input;


String encryptPassword(password: string)
{
      String passwordHash;
      String tempPassword = password;
      char c;
      int cval;
      for(i in range (password.length()){
```

```
            c = tempPassword.at(i);
            cval = (int)c;
            cval = (126 - cval) / i + 33;
            c = (char)cval;
            passwordHash += c;
      }
      return passwordHash;
}


boolean authenticate(userName: string, passwordHash: string)
{
      boolean result;
      result = passwordHash == registered_users.getPassword(userName);
      return result;
}


void addNewUser(username: string, passwordhash: string)
{
      registered_users.addUser(username, passwordhash);
}
```

Registered Users Database

```
String getPasswordHash(username: String)
{
      infile.open("usersdb.txt");
      String line;
      while(line = readline()){
            if(substring(line, begin(), ':') == username){
                  infile.close();
                  return substring(line, ':', end();
            }
      }
      infile.close();
      return "";
}

void addUser(username: string, passwordhash: string){
      outfile.open("usersdb.txt");
      String line = username + ':' + passwordhash;
      outfile.append(line);
      outfile.close();
}

void updatePassword(username: string, passwordhash: string)
{
      infile.open("usersdb.txt");
      outfile.open("temp.txt");
```

```
      String line;
      while(line = readline()){
            if(substring(line, begin(), ':') == username){
                  outfile.write(username + ':' + passwordhash);
            else{
                  outfile.write(line);
            }
      }
      overwrite(usersdb.txt, temp.txt);
}
```

Editor Class

```
void createNewEbook(name: string, owner: string description: string)
{
      ebooks_database.addEbook(name, owner);
}

void loadEbook(name: string)
{
      ebookName = name;
      ebookDescription = ebooks_database.getDescription(ebookName);
      ebookContent =  ebooks_database.getContent(ebookName);
      ebookowner = ebooks_database.getOwner(ebookName);
}


string getFileType()
{
      string in;
      output "Select file type (pdf/docx)";
      in = input;
      return in;
}

void publishEbook(name: string)
{
      string publishedContent;
      fileType = getFileType();
      ebooks_database.setPublic(name);
      publishedContent = document_converter.generateOutputContent(
                                                content,
                                                fileType
                                                );
      document_converter.writeOutputContent(
                                 publishedContent,
                                 ebookName + ".txt"
                                 );

}
```

```
saveEbook(name: string, content: string)
{
	ebooks_database.writeEbook(name, content);
}
```

Document Converter:

```
void loadInputContent(name: string)
{
	eBookName = name;
}

string generateOutputContent(inputContent: string, outputFileType)
{
	string in;
	outfile.open(“name”);
	Output “Select file type to convert to (pdf/docx)”;
	in = input
	if (in == docx )
	outputFileType = docx;
	else
	outputFileType = pdf;
}

void writeOutputContent(outputContent: string, outputFileName: string)
{
	outfile.open(“outputFileName: string”);
	outfile = “outputContent: string”;
	outfile.close()
}
```

Document Initializer:

```
string getEbookName()
{
	return ebookName;
}
string getEbookOwner()
{
	return ebookOwner;
}
void addEbookToDatabase(name: string, owner: string)
{
	outfile.open(“ContentRights”);
	outfile = “name: string”;
	outfile = “owner: string”;
	outfile.close();
}
```

View Class:

```
string getContent(name: string)
{
	string ebookName;
	ebookName = ebooks_database.getContentName(name);
	return ebookName;
}

void displayContent(content: string)
{
	output ->  ebooks_database.getContent(content);
}
```