# IDS517 2015 FALL PROJECT Group – f15g114 PROJECT DOCUMENT

### **Group Members**

- 1. Nitheen Nallay Palli
- 2. Vinodh Sankaran
- 3. Asawari Shekatkar
- 4. Gurneha Naggi

# Contents

Project Objective	3
Project Flow Diagram	
Project Functional Overview	
Class Diagram	7
Database Schema Diagram	8

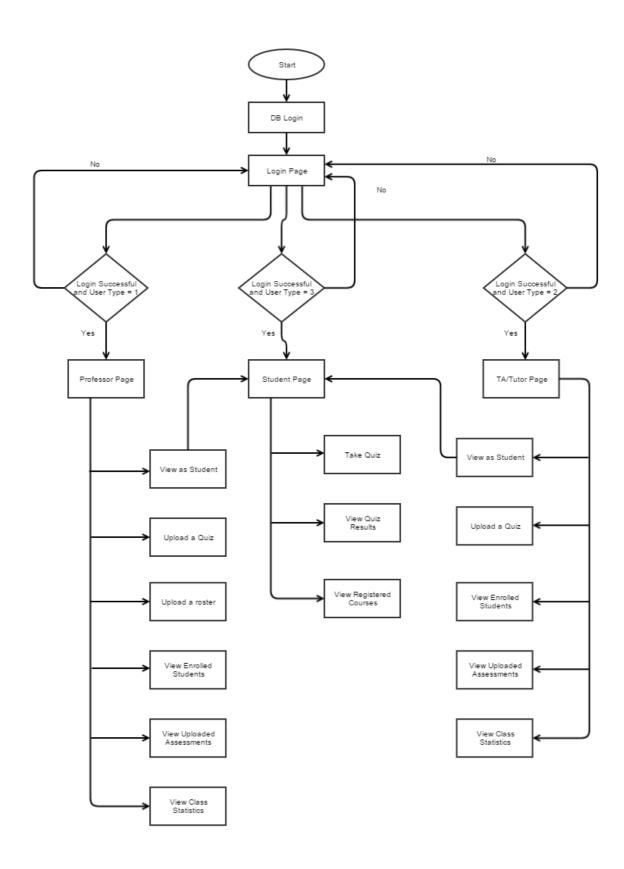
# **Project Objective**

The objective of this project is to develop an online system to facilitate quiz and exam administration and grading for quantitative statistics and math courses. The application allows the professors to upload assessments for the courses they handle. The students respond to the assessment and are provided with real-time course grades helping them know how they are performing in each course. The application also consist of the historical performance of the students helping the professors in assessing the student's performance through statistical analysis and plan for scope of improvement.

The project is designed using JSF framework. This project follows MVC model. The document contains the following.

- Project Flow Diagram
- > Class Diagram
- Database Schema Diagram

# Project Flow Diagram



## **Project Functional Overview**

The application has the following functions.

#### **User Login:-**

The application can be used by different users such as professor/instructor, TA/Tutor and student. When a user logs into the application, the user is validated based on the user type value in the database and provide only restricted access to each user based on the user type. The login timestamp and IP of each user is logged into the database for security purpose.

Each user should be able to view the last login in the bottom of the welcome page. In case of first time login, default value will be displayed.

#### **Student Functions: -**

#### **Take Up Quiz**

A student can take up the quiz once it is uploaded by the professor/TA. The responses provided by the students are stored into the response table in the database. The responses are evaluated by comparing the student response with the actual answers based on the tolerance level and graded accordingly.

#### **View Quiz Results**

The students can view their quiz results once graded. In addition in viewing their quiz grades, they can also review their answers and compare them with the actual answers.

#### **View Registered Courses**

The students can view the courses in which they are enrolled. This will help them in keeping track of all the announcements happening in each courses and reviewing the answers to the assessments.

#### Professor/TA/Tutor Functions: -

#### **Upload Roster**

Professor uploads the roster file consisting of the student information who are newly enrolled in a course. The roster file consists of student's first name, last name, net id and course that they are enrolled. The roster file will be in .csv format.

#### **View as Student**

Professor/TA/Tutor will be able to view the application as a student. It helps them in checking the student functionalities like taking up an assessment, view their grades, etc.

#### **Upload a Quiz**

Professor/TA/Tutor uploads the assessment file consisting of the quiz information. The assessment file will be in .csv format. The assessment file consists of assessment questions along with the actual answer and tolerance level. Each time an assessment file is uploaded, a new table will be created in the database for the assessment. Also, an entry will be made in assessments table in order to keep track of the user who has uploaded the file along with the course id.

#### **View Enrolled Students**

Professor/TA/Tutor would be able to view the student details. It helps them to generate report containing results of numeric and graphic analysis such as descriptive statistics of the class.

#### **View Student Grades**

Professor/TA/Tutor can view the student grades along with their responses in each assessment. Here, the professor will be able to provide the feedback based on the performance of the student in the assessment.

#### **View Uploaded Assessments**

Professor/TA/Tutor can view the assessment which they have uploaded in order to verify the assessment file.

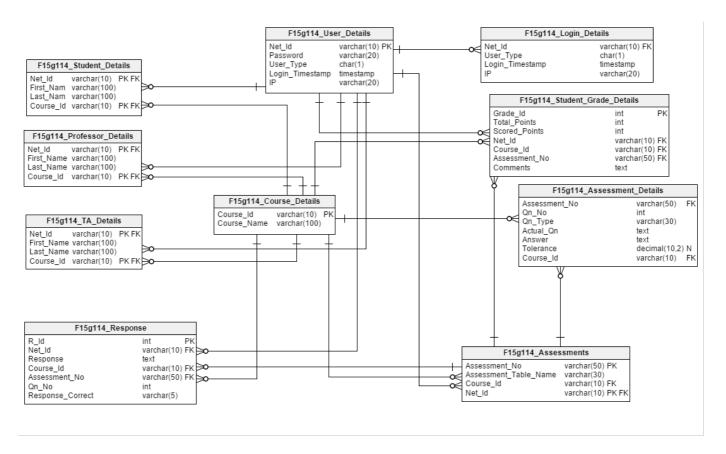
#### **View Class Statistics**

Professor/TA/Tutor can view the class statistics to assess the student's performance in each assessment. It enables for graphic analysis capability such as histogram, bar chart and pie chart distribution of scores. The instructor plan for scope for improvement in the course structure as well as for the student based on the score distribution.

# Class Diagram

ProfessorBeam4tttpServlefResponseedu uici is model     uploadedFile uploadedFile     fileLabet String     fileNams. String     inBeNams. String     inBeContentType: String     name: String     incomments: String     incomments: String	G StudentBean edu.uic.ids.model a netid: String		uestionBean				<ul> <li>DataValuesBea</li> </ul>			
o fletabel: String o fletame: String o fletcontentType: String o name: String o comments: String			edu. uic. ids. model		● LoginDao edu.uio.ids.model		edu.uic.ids.DbConnection		edu.uic.ids.model	
a fleName: String a fleContentType: String a name: String a comments: String		△ assessment_no: String		△ loginBean: I	_oginBean		△ userName: String		a firstName: String	
a fileContentType: String name: String comments: String	a firstName; String	△ question_no: int		△ connection:			△ password: String		a lastName: String	
name: String	a lastName: String	△ question_type: String					△ dbms: String		netld: String	
comments: String	a courseID: String	△ actual_question: String			△ ec: ExternalContext		△ dbmsHost: String		password: String	
	a assessment: String	△ answer: String		A session: Ht	session: HttpSession		△ schema: String		a role: String	
	answer: String comments: String	△ tolerance: double		△ userTypeld:	userTypeld: int		©DataValuesBean()		a course: String	
names: ArrayList <string> student_name: String</string>	n max_score: int	A courseID: String		©LoginDao()	LoginDao()		o init():void		course_name: String	
student_list: ArrayList <string></string>	p actual_score: int	△ max_points: int  △ actual_points: int		<ul><li>checkLogin</li></ul>	checkLogin(String,String):int		⊕ getUserName():String		△ courseName: ArrayList <string>  △ conn: Connection</string>	
assessment: ArrayList <questionbean></questionbean>	grade_id: int	△ netid: String					<ul><li>setUserName(String)</li></ul>		a tableName: String	
roster: ArrayList <studentbean></studentbean>	△ conn: Connection	△ response: String					getPassword():String	The same of the sa	△ tables: ArrayList <string></string>	
△ statement: Statement	△ resultset: ResultSet	△ response_correct: Strir	ng				<ul> <li>setPassword(String)</li> <li>qetDbms():String</li> </ul>	void	△ displayTables: boolean	
a resultset: ResultSet	△ statement: Statement	@ getAssessment_no():S					setDbms(String):void		dynamicList: List <list<string>&gt;</list<string>	
a conn: Connection	△ assessments: ArrayList <string></string>	setAssessment_no(Str					<ul> <li>getDbmsHost():String</li> </ul>		△ dynamicHeaders: ArrayList <string></string>	
displayAssessment: boolean	△ displayQuiz: boolean	getQuestion_no():int					<ul> <li>setDbmsHost(String):</li> </ul>		u dynamicDataTableGroup: HtmlPanelGroup	
display: boolean	△ displayList: boolean	setQuestion_no(int):voi	oid		I I		getSchema():String		o <sup>C</sup> TutorBean()	
∆ displayRoster: boolean	△ instructorAccess: boolean	getQuestion_type():Stri	10.0			setSchema(String):void		id	createTables():String	
△ displayStudent: boolean	△ questions: ArrayList <questionbean></questionbean>	setQuestion_type(Strin				checkDbValues():String			dropTables():String	
br: BufferedReader	△ response: ArrayList <string></string>	getActual_question():S	tring						displayTable():String	
NEW_LINE_SEPARATOR: String	A actual_answers: ArrayList <string></string>	setActual_question(Str	ing):void				< <java class="">&gt;</java>		dropAllTables():String	
a ec: ExternalContext	△ assessment_view: ArrayList <questionbean></questionbean>	getAnswer():String					DbConnection		viewTables():String	
A session: HttpSession	Δ response_type: ArrayList <string></string>	setAnswer(String):voic			-		du.uic.ids.DbConnection	4	viewTables_assessments():String	
sb: StudentBean	△ tolerance: ArrayList <string></string>	getTolerance():double	0.000			Surl: Stri			<ul> <li>createTableLoad():String</li> </ul>	
ProfessorBean()	△ ec: ExternalContext  △ session: HttpSession	setTolerance(double):v	oid				ame: String		o registerUser():String	
processFileUpload():String	1,000,000	getCourseID():String	V.				me: String		loadRegister():String	
uploadProcess():String	loadAssessments():String	setCourseID(String):voi	id		-	_	ord: String		newCourse():String	
viewAssessments():String	processAssessments():String	getMax_points():int				DbConr			processDynamicTables():String	
processViewAssessments():String	evaluateAssessment():String	setMax_points(int):void					nection():Connection		■ createValueExpression(String,Class ):ValueExpress	
viewAssessmentsRoster():String	isNumeric(String):boolean	getActual_points():int			•	CloseCo	onnection(Connection):boo	lean	clearDynamicList():String	
fetchStudentList():String	loadGrades():String	setActual_points(int):vo	oid				< <java class="">&gt;</java>		getTableName():String	
displayRosterReset():String fetchStudentAssessment():String	processGrades():String     getScores(Connection,String,String,String):void	<ul> <li>getNetid():String</li> <li>setNetid(String):void</li> </ul>					⊕ LoginBean		setTableName(String):void	
processDeleteAssessments():String	studentView():String	getResponse():String					edu.uic.ids.model		getTables():ArrayList <string></string>	
saveComments():String	professorView():String	getResponse_correct()	to Otelia a			ΔU	serName: String		setTables(ArrayList <string>):void</string>	
processRosterUpload():String	fetchStudentAssessment():String		(i)			100000	assword: String		isDisplayTables():boolean	
viewRoster():String	init():void	© QuestionBean(int,String	QuestionBean(String,int,String,String,String,double,String)				stLogin: String		setDisplayTables(boolean):void     setEiratNama():Pfring	
classGrades():String	getGrade_id():int	© QuestionBean(String,St				△ fi	rstName: String		getFirstName():String     setFirstName(String):void	
processClassGrades():String	getNetid():String	© QuestionBean(int,String				A lo	ginCheck: boolean		getLastName():String	
processFileDownload():String	setNetid(String):void	- adolesion beautiful carried	gioting (string)			△ lo	ginURL: String		setLastName(String):void	
getUploadedFile():UploadedFile	getFirstName():String		Library Olevania		1	△ e	c: ExternalContext		getNetId():String	
setUploadedFile(UploadedFile):void	<ul><li>setFirstName(String):void</li></ul>		<< Java Class>>			A S	ession: HttpSession		setNetId(String):void	
getFileLabel():String	getLastName():String		edu.uic.ids.model			e <sup>C</sup> L	oginBean()		getPassword():String	
setFileLabel(String):void	setLastName(String):void		△ ec: ExternalContext			1000000	etUserName():String		setPassword(String):void	
getName():String	getCourseID():String		△ session: HttpSession				etUserName(String):void		getRole():String	
setName(String):void	<ul><li>setCourselD(String):void</li></ul>		A resultset: ResultSet		7/3		etPassword():String		setRole(String):void	
getNames():ArrayList <string></string>	@ getAssessment():String		△ statement: Statement		setPassword(String):vo		etPassword(String):void		getCourse():String	
setNames(ArrayList <string>):void</string>	setAssessment(String):void		△ conn: Connection		getLastLogin():String		etLastLogin():String		setCourse(String):void	
getAssessment():ArrayList <questionbean></questionbean>	getAssessments():ArrayList <string></string>		△ display: boolean			setLastLogin(String):void			getCourseName():ArrayList <string></string>	
setAssessment(ArrayList <questionbean>):void</questionbean>	<ul> <li>setAssessments(ArrayList<string>):void</string></li> </ul>		Δ displayAssessment: boolean			getFirstName():String			setCourseName(ArrayList <string>):void</string>	
isDisplayAssessment():boolean	isDisplayQuiz():boolean		A assessmentList: ArrayList <string></string>			setFirstName(String):void			getCourse_name():String	
setDisplayAssessment(boolean):void	setDisplayQuiz(boolean):void     isPinglayd (at/) boolean		△ assessment: String			18.900	alidate():String		setCourse_name(String):void	
getRoster():ArrayList <studentbean></studentbean>	isDisplayList():boolean     actDisplayList():boolean		Δ statDesc: String			⊚ lo	gout():String		getDynamicList():List <list<string>&gt;</list<string>	
setRoster(ArrayList <studentbean>):void</studentbean>	setDisplayList(boolean):void     netOuestings():Arrayl ists(QuestingSean)		△ statValue: double						setDynamicList(List <list<string>&gt;):void</list<string>	
isDisplayRoster():boolean	getQuestions():ArrayList <questionbean>     setQuestions(ArrayList<questionbean>):yold</questionbean></questionbean>		△ statistics: ArrayList <chartbean< td=""><td>&gt; :</td><td></td><td></td><td></td><td></td><td>getDynamicHeaders():ArrayList<string></string></td></chartbean<>	> :					getDynamicHeaders():ArrayList <string></string>	
setDisplayRoster(boolean):void getStudent_name():String	setQuestions(ArrayList <questionbean>):void     getAnswer():String</questionbean>		processBarChart():String		]				setDynamicHeaders(ArrayList <string>):void</string>	
setStudent_name():String setStudent_name(String):void	setAnswer().String     setAnswer(String):void		o createBarChart(String,String,Ot					getDynamicDataTableGroup():HtmlPanelGroup		
getStudent_list():ArrayList <string></string>	getMax_score():int		processTimeSeriesChart():Strin				3	<ul> <li>setDynamicDataTableGroup(HtmlPanelGroup):void</li> </ul>		
setStudent_list(ArrayList <string>):void</string>	setMax_score(int):void									
isDisplayStudent():boolean	getActual_score():int		descriptiveStatsLoad():String							
setDisplayStudent().buolean):void	setActual_score(int):void		descriptiveStats():String							
getComments():String	getResponse():ArrayList <string></string>		addValue(double,String):void							
isDisplay():boolean	setResponse(ArrayList <string>):void</string>		<ul><li>isDisplayAssessment():boolean</li></ul>							
setDisplay(boolean):void	getActual_answers():ArrayList <string></string>		<ul> <li>setDisplayAssessment(boolean):void</li> </ul>							
setComments(String):void	setActual_answers(ArrayList <string>):void</string>		<pre>     getAssessmentList():ArrayList<string> </string></pre>							
	⊚ getComments():String		setAssessmentList(ArrayList<	String>):void						
	setComments(String):void		<ul><li>isDisplay():boolean</li></ul>							
	<ul><li>isInstructorAccess():boolean</li></ul>		setDisplay(boolean):void							
	<ul> <li>setInstructorAccess(boolean):void</li> </ul>		getAssessment():String							
	getAssessment_view():ArrayList <questionbean></questionbean>		setAssessment(String):void	2000						
	setAssessment_view(ArrayList <questionbean>):void</questionbean>		getStatistics():ArrayList <charte< td=""><td></td><td></td><td></td><td></td><td></td><td></td></charte<>							
	©StudentBean(String,String,String,String)		setStatistics(ArrayList <chartbe< td=""><td>an&gt;):void</td><td></td><td></td><td></td><td></td><td></td></chartbe<>	an>):void						
	©StudentBean()		getStatDesc():String     setStatDesc(String):void							
	© StudentBean(String,String,int,int,String)									
			<ul> <li>getStatValue():double</li> <li>setStatValue(double):void</li> </ul>							
			ChartBean(String,double)							
			ChartBean()							

## Database Schema Diagram



#### F15g114\_User\_Details: -

The F15g114\_user\_details table stores the login information of all the users who login into the application. It stores the user id of the user who has logged in, type of the user (restrict access based on the user type) and IP address of the user.

#### F15g114\_Login\_Details: -

The F15g114\_Login\_Details table stores the login information of all the users who are logging into the application. It stores the history of user login details along with the IP address and login time.

#### F15g114\_Student\_Details: -

The F15g114\_student\_details table has all the student information such as their first name, last name and net id. This table allows the student to check and view their profile information. This table refers the course details table and gives information on the courses enrolled by the student.

#### F15g114\_Professor\_Details: -

The F15g114\_professor\_details table has all the professor information such as their first name, last name and net id. This table allows the professor to check and view their profile information. This table refers the course details table and gives information on the courses taught by the professor.

#### F15g114\_TA\_Details: -

The F15g114\_ta\_details table has all the teaching assistant information such as their first name, last name and

net id. This table allows the teaching assistant to check and view their profile information. This table refers the course details table and gives information on the courses enrolled by the teaching assistant.

#### F15g114\_Course\_Details: -

The F15g114\_course\_details table has the details on the total number of courses available in the institution. A student can register in a course only if the course is available in this table.

#### F15g114\_Assessments: -

The F15g114\_assessments table consists of the assessment details such as the assessment number, table name of the new assessment uploaded, uploaded professor net id and the course id for which the assessment has been uploaded. It keeps tracks of all the uploaded assessments.

#### F15g114\_Assessment\_Details: -

The F15g114\_assessment\_details table will be created each time the professor/TA uploads an assessment into the application. It consist of the assessment questions along with the actual answer and the tolerance level.

#### F15g114\_Response:-

The F15g114\_response table stores the response provided by the students for each assessments. It has the student net id who provide the response along with the course id and assessment number.

#### F15g114\_Student\_Grade\_Details: -

The F15g114\_student\_grade\_details table has the grade details for each of the student. It consists of the grades along with the course id and assessment number of the graded course with professor comments.