

2CPI



**Rapport de projet :**

**Sous la supervision de :**

\*GHEID ZAKARIA

**Réalisé par :**

\*Bendahmane Hamza

\*Bousaada Moustafa

\*Manaa Anouar

\*Larbaoui Nour El Imène

\*Boufeldja Manel

\*Bouziane Djihane

***Table des matières :***

*Introduction :…………………………………………………………………………………………………………………………4*

*Les objectifs :…………………………………………………………………………………………………………………………4*

*Le domaine d’utilisation :……………………………………………………………………………….……………………..4*

*Le principe du fonctionnement :…………………………………………………………………………………………….4*

*Les outils et langages utilisés :……………………………………………………………………………………………….5*

*Les grandes fonctionnalités :………………………………………………………………………….……………………..6*

*1-partage des tutoriels et des cours :……………………………………………………………………………………..6*

*2-Echange des idées et des articles :………………………………………………………………………………………6*

*Conception :…………………………………………………………………………………………………………………………..6*

*1-Article :………………………………………………………………………………………………………………………..6*

*Ajouter article :………………………………….……………………………………………………………………..7*

*Cmmenter l’article :………………………………………….……………………………………………………….8*

*Meilleurs articles :………………………………………………….……………………………………..………….9*

*2-Cours :……………………………………………………………………………………………….………………………10*

*Tutoriels :……………………………………………………………………………….………………………..…..10*

*Résumés :…………………………………………………………………………………….………………………..17*

*Firebase :…………………………………………………………………………………………………………………………….17*

*Authentification :…………………………………………………………………………………………………………..17*

*Authentification avec google :……………………………………………………….………………………..……17*

*Méthodes de connexion :…………………………………………………………….………………………………18*

*Vérification des émails :………………………………………………………………….……………………………19*

*Liste des utilisateurs :………………………………………………………………………….…………………..…..19*

*Cloud Firestore :………………………………………………………………………………………………………………20*

*Cloud storage :…………………………………………………………………………………………………………………20*

*Realtime database :………………………………………………………………………………………………………….21*

*Les permissions :………………………………………………………………………………………………………………….21*

*Aperçu sur notre application :……………………………………………………………………………………………..22*

***Remerciements***

*Avant tout, nous remercions Allah tout puissant, pour nous avoir donné la force et la persévérance de mener à bien ce projet.*

*Nous remercions notre encadreur : Mr GHEID ZAKARIA pour ses précieux conseils et pour nous avoir accompagné durant la réalisation, et les membres du jury pour l’honneur qu’ils nous accordent en acceptant d’évaluer notre travail.*

*Nous sommes également très reconnaissants envers les amis et collègues qui nous ont été un appui moral et intellectuel tout au long du déroulement de ce projet.*

*Finalement, nous remercions tous ceux et celles qui ont contribué de près ou de loin à l'aboutissement de cet humble travail.*

***INTRODUCTION :***

C’est grâce à ce mini projet que nous avons eu l’opportunité de cumuler les connaissances avec celles de la pratique, dans le cadre de la 2ème Année du cycle préparatoire de l’ESI SBA ceci permet également de rentrer dans la vie universitaire quotidienne de l’étudiant et essayer de la faciliter.

Notre application ESI better est conçue pour rassembler des tâches importantes dont l’étudiant d’ESI a besoin dans ses études, comme l’échange des idées et des articles entre les étudiants, partage des cours et des tutoriels et tout ce qui utile, et d’autres fonctions intéressantes.

***LES OBJECTIFS*** :

* Partage des tutoriels, des cours et des résumés.
* Echange des idées et des articles entre les étudiants.
* Faciliter les études et gagne du temps.
* Obtenir des ressources d’éducation gratuit.
* Développer et enrichir les connaissances de l’étudiant.
* Mettre les étudiants à jour avec leurs idées et déterminer les meilleurs.

***LE DOMAINE D’UTILISATION :***

Cette application est destinée seulement aux étudiants de 1ère et 2ème année du cycle préparatoire d’ESI SBA.

***PRINCIPE DE FONCTIONNEMENT :***

ESI better est une application mobile Android, pour la fonctionner il suffit que l’étudiant créer un compte à l’aide de son adresse email et bénéficier les fonctionnalités.

***LES OUTILS ET LANGAGES UTILISES :***

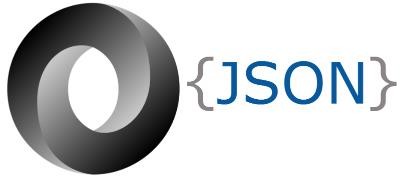
**Android Studio** est un environnement de développement pour développer des applications mobiles Android. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il peut êtretéléchargé sous les systèmes d'exploitation Windows, macOS et Linux.

**L'Extensible Markup Language**, généralement appelé XMLnote 1, « langage de balisage extensible1 » en français, est un métalangage informatique de balisage générique qui est un sous-ensemble du Standard Generalized Markup Language (SGML). Sa syntaxe est dite « extensible » car elle permet de définir différents langages avec chacun leur vocabulaire et leur grammaire, comme XHTML, XSLT, RSS, SVG… Elle est reconnaissable par son usage des chevrons (<, >) encadrant les noms des balises. L'objectif initial de XML est de faciliter l'échange automatisé de contenus complexes (arbres, texte enrichi, etc.) entre systèmes d'informations hétérogènes (interopérabilité). Avec ses outils et langages associés.

**Java**est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld.

La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java.

La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d’exploitation tels que Unix, Windows, Mac OS ou GNU/Linux, avec peu ou pas de modifications, mais qui ontl'inconvénient d'être plus lourd à l'exécution (en mémoire et en temps processeur) à cause de sa machine virtuelle. Pour cela, divers plateformes et frameworks associés visent à guider, sinon garantir, cette portabilité des applications développées en Java.

JSON (JavaScript Object Notation) est un format d'échange de données en texte lisible. Il est utilisé pour représenter des structures de données et des objets simples dans un code qui repose sur un navigateur Web.

**Firebase** est un ensemble de services d'hébergement pour n'importe quel type d'application (Android, iOS, JavaScript, Node.js, Java, Unity, PHP, C++ ...). Il propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification sociale (Google, Facebook, Twitter et Github), et des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel. Lee et par James Templin, le service est racheté par Google en octobre 2014.

L'interface de ligne de commande Firebase fournit une variété d'outils pour la gestion, l'affichage et le déploiement dans les projets Firebase on a l’utiliser specifiquement pur déploier les functions pour le backend.

**Adobe XD** est une solution d'UX/UI design complète pour la conception de sites web, d'applications mobiles, etc. Créez, prototypez et partagez, le tout depuis Adobe XD.

Alliant rapidité, précision et qualité, Adobe XD permet aux designers de modifier et partager facilement des prototypes interactifs avec collaborateurs et réviseurs sur l'ensemble des appareils et plates-formes, dont Windows, Mac, iOS et Android.

**Node JS « JavaScript »**est un langage de programmation de scripts principalement employé dans les pages web interactives mais aussi pour les serveurs avec l'utilisation (par exemple) de Node.js.

**Photoshop** est un [logiciel](https://fr.wikipedia.org/wiki/Logiciel) de [retouche](https://fr.wikipedia.org/wiki/Retouche_d%27image), de [traitement](https://fr.wikipedia.org/wiki/Traitement_d%27images) et de [dessin assisté par ordinateur,](https://fr.wikipedia.org/wiki/Dessin_assist%C3%A9_par_ordinateur) Édité par [Adobe](https://fr.wikipedia.org/wiki/Adobe_Systems), il est principalement utilisé pour le traitement des [photographiesnumériques](https://fr.wikipedia.org/wiki/Photographie_num%C3%A9rique), mais sert également à la création [*ex nihilo*](https://fr.wikipedia.org/wiki/Ex_nihilo) *d’images.*

Iltravailleessentiellementsur[imagesmatricielles](https://fr.wikipedia.org/wiki/Image_matricielle)carlesimagessontconstituéesd’une grilledepointsappelés[pixels.](https://fr.wikipedia.org/wiki/Pixel)L’intérêtdecesimagesestdereproduiredesgradationssubtilesdecouleurs

***LES GRANDES FONCTIONNALITES :***

**1-Partage des tutoriels et des cours*:***

-Ajouter des tutoriels et des cours intéressants pour tous les modules.

-Ajouter des vidéos éducatifs.

-Visionnement et chargement des vidéos et des tutoriels.

**2-Echange des idées et des articles :**

-Ajouter des idées et des articles.

-Ajouter des commentaires sur les idées partagées.

-like et dislike et évaluer les meilleurs idées.

***CONCEPTION :***

* ***BDD (NOSQL Firebase Database) :***
* Les bases de données NoSQL sont de plus en plus utilisées maissavez-vous vraimentcequ’elles renferment ? Prenons les exemples de Datastax (Cassandra) et MongoDB qui figurent parmi les solutions NoSQL les plus utilisées dumarché.

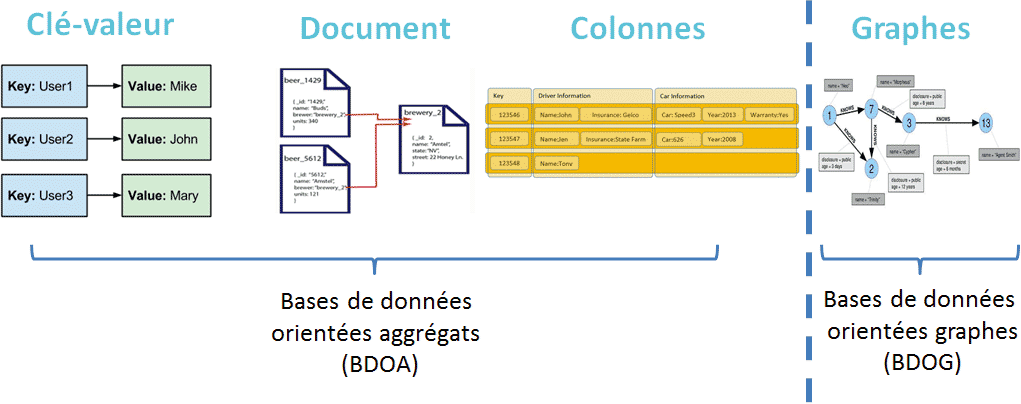
**A quoi sert une base NoSQL ?**

Lesbasesrelationnellesontété«inventées»parEdgarF.Codden1970.Al’époque,cefutune approche révolutionnaire (modèle relationnel des données, algèbre de relations,sélections,

projections,jointures,etc…).Oraclefutlapremièresociétéàcommercialiserunproduitconformeàce concept,dès1977,aveclesuccèsquel’onconnait.Cesdernièresannées,plusieursavisnuancésont été émis concernant le modèle relationnel:

* Difficultéspoursauveretrécupérerdesgrappesd’objetsavecdesprogrammesécritsenJavaouC#... (on parle de « mismatchd’impédance)
* Difficultés pour satisfaire les besoins des applications Web à grande échelle (nouveauxbesoins métiers, nouvelles contraintestechniques)

Voici les 3 aspects qui ont motivé la nouvelle vague de bases NoSQL (Not Only SQL...) :

* LeVOLUMEdesdonnéescrééesdoubletousles2ans.IDCestimequ’en2020levolumeatteindra44 Zettabytes (1 ZB = 1 milliards deterabytes)
* La VARIÉTÉ des types de données créées(Smartphones)
* La VÉLOCITÉ avec laquelle les données changent est également très importante (Internet of Things) En fait, le terme base NoSQL définit une nouvelle génération de produits qui ne suivent pas le modèle relationnel.Maisl’architecturedecesproduitsvariebeaucoupentreeux.
* **Les types de bases de données NoSQL :**
* Clé- valeur, document, colonnes et graphes sont les 4 types de bases de données NoSQL. Etudions d’un peu plus près chacun de ces types.******

Notre type choisi est le type “Clé-valeur” ce qu’on appelle en Firebase « Firebase Relatime Database » et on a utilisée aussi FirestoreFirebase qu’utilise les Documents .

1 /**Article :** Cette partie contient tout ce qui concerne les articles, qui est composée de trois principales fonctionnalités :

**Articles**

Commenter l’article

Meilleur article

Ajouter l’article

Récupérer tous les infos sur l’article

Evaluer les meilleurs articles

Peut like ou dislike et répondre sur le commentaire

**1 -1/**Ajouter Article :

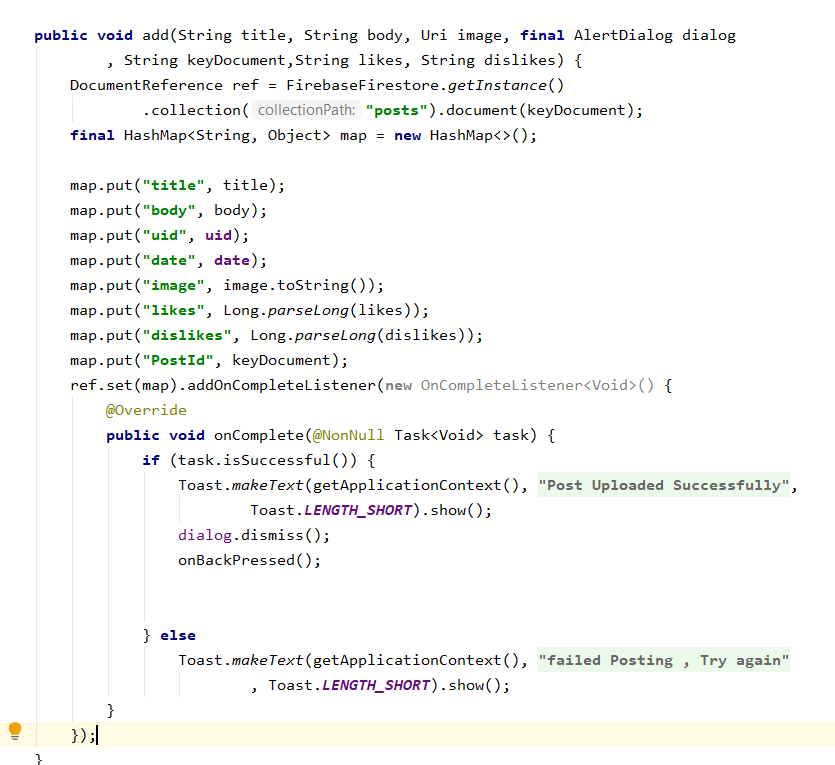
La classe Article\_item.java permet de récupérer tous les informations sur l’utilisateur et son article publié :

**Class Article\_item.java :**



Et la classe Add\_Article.java permet d’ajouter l’article et sauvegarder les informations récupérées.

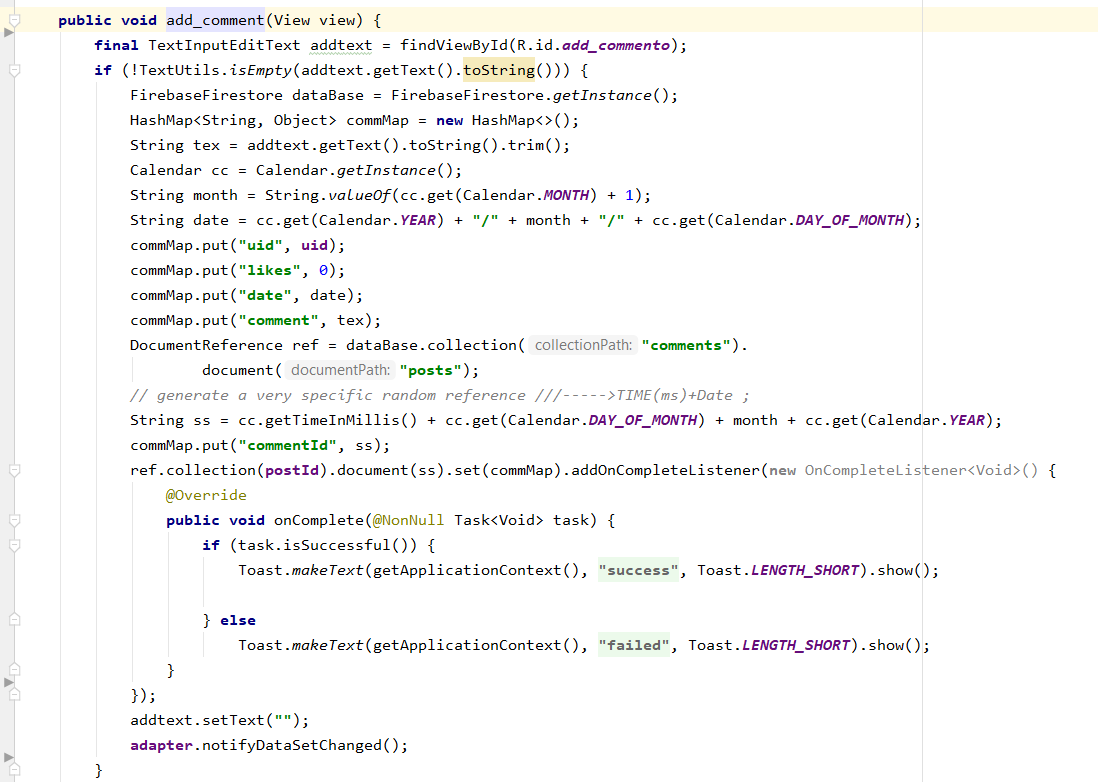
**Class\_Add\_Article.java :**



**1-2/**Commenter l’article :

La classe Add\_Comment.java aider à ajouter des commentaires sur un article :

**Class\_Add\_Comment.java :**



**1-3/**Meilleurs articles :

Les articles sont classifiés par rapport le nombre des likes obtenus , et la classe bestOf.java réalise cette fonctionnalité :

**Class\_bestOf.java :**



**2/Cours :**

Cours

Tutoriels

Résumés

**2-1/**Tutoriels :

1. Nous créons la classe Add\_tutorial qui est constitués de tout les nécessaires traitements pour ajouter un tutoriel :

**Class Add\_Tutorial.java :**

|  |  |
| --- | --- |
| **Code** | **Commentaire** |
| **public class** Add\_Tutorial **extends** AppCompatActivity { **public final** String **uid** = FirebaseAuth.*getInstance*().getCurrentUser().getUid(); **public final** CollectionReference **reference** = FirebaseFirestore.*getInstance*()  .collection(**"posts"**); **public** Uri **image\_art**, **file\_art**; **final** Calendar **cc** = Calendar.*getInstance*(); **final** String **month** = String.*valueOf*(**cc**.get(Calendar.***MONTH***) + 1); **final** String **date** = **cc**.get(Calendar.***YEAR***) + **"/"** + **month** + **"/"** + **cc**.get(Calendar.***DAY\_OF\_MONTH***);  String **titleArt**;  TextView **post\_article**;  String **dislikes** = **"0"**;  Spinner **yearSpin**, **ModuleSpin**;  String **Module**; **int Type**, **Year**;  StorageReference **images\_url**;  ImageView **show\_picked\_image**;  String **BodyArt**; **boolean haveChoosedTheType**, **haveChoosedTheModule**;  String **likes** = **"0"**;  EditText **TitleBody**;  String **typeof** = **"add"**; **boolean Done** = **false**;  CircleImageView **photo**;  ArrayAdapter<CharSequence>**adapterModule**;  @Override **protected void** onCreate(Bundle savedInstanceState) { **super**.onCreate(savedInstanceState);  setContentView(R.layout.***courses\_add\_tutorial***) ;**typeof** = getIntent().getExtras().getString(**"type"**); **post\_article** = findViewById(R.id.***post\_Tutorial***); **photo** = findViewById(R.id.***picture***); **show\_picked\_image** = findViewById(R.id.***imageView6***); **TitleBody** = findViewById(R.id.***Title***); **images\_url** = FirebaseStorage.*getInstance*().getReference(**"Images/"** + **uid**);**ModuleSpin** = findViewById(R.id.***ModuleSpin***); **yearSpin** = findViewById(R.id.***yearSpin***); | Déclaration des variables pour récupérer tous les informations sur le tutoriel.  Initialisation des variables. |
| findViewById(R.id.***Add***).setOnClickListener(**new** View.OnClickListener()   { @Override **public void** onClick(View v) {  pick\_Image(**null**);}  }); findViewById(R.id.***imageView2***).setOnClickListener(**new** View.OnClickListener() {  @Override **public void** onClick(View v) {  pick\_Video(**null**);  } }); | Choisir une image ou /et sélectionner un fichier. |
| **yearSpin**.setEnabled(**true**); **yearSpin**.setSelection(0);**final** ArrayAdapter<CharSequence> adapterYear = ArrayAdapter.*createFromResource*(**this**,  R.array.***year***, android.R.layout.***simple\_spinner\_item***);  adapterYear.setDropDownViewResource(android.R.layout.***simple\_spinner\_dropdown\_item***); **yearSpin**.setAdapter(adapterYear); **yearSpin**.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  @Override **public void** onItemSelected(AdapterView<?> adapterView, View view, **int** i, **long** l) { **Year** = i; **haveChoosedTheType** = !(**Type**==0); **haveChoosedTheModule** = !(**Year**==0); **if** ( **haveChoosedTheModule** ) **ModuleSpin**.setEnabled(**true**); **else** { **ModuleSpin**.setEnabled(**false**); **Done** = **false**;  } **if** (**Year**==1)  { **adapterModule** = ArrayAdapter.*createFromResource*(Add\_Tutorial.**this**,  R.array.***first\_modules\_titles***, android.R.layout.***simple\_spinner\_item***); **adapterModule**.setDropDownViewResource(android.R.layout.***simple\_spinner\_dropdown\_item***);  **ModuleSpin**.setAdapter(**adapterModule**);  **ModuleSpin**.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  @Override **public void** onItemSelected(AdapterView<?> adapterView, View view, **int** i, **long** l) { **Module** = **ModuleSpin**.getSelectedItem().toString(); **Done** = **true**;  }  @Override **public void** onNothingSelected(AdapterView<?> adapterView) {   }  });   } **else if**(**Year**==2){ **adapterModule** = ArrayAdapter.*createFromResource*(Add\_Tutorial.**this**,  R.array.***second\_modules\_titles***, android.R.layout.***simple\_spinner\_item***);  **adapterModule**.setDropDownViewResource(android.R.layout.***simple\_spinner\_dropdown\_item***); **ModuleSpin**.setAdapter(**adapterModule**);  **ModuleSpin**.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  @Override **public void** onItemSelected(AdapterView<?> adapterView, View view, **int** i, **long** l) { **Module** = **ModuleSpin**.getSelectedItem().toString(); **Done** = **true**;  }   @Override **public void** onNothingSelected(AdapterView<?> adapterView) {   }  });  } | Traitement pour déterminer l’année correspondante de tutoriel publié.  **1CPI**  **2CPI** |
| **post\_article.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  Bundle bundle = getIntent().getExtras(); String title = TitleBody.getText().toString().trim() ;  boolean imageIsSET = image\_art != null;  boolean fileIsSet = file\_art!=null;**  **if (TextUtils.*isEmpty*(title))  Toast.*makeText*(Add\_Tutorial.this, R.string.*title\_shou\_em*, Toast.*LENGTH\_SHORT*).show();  else if (!imageIsSET)  Toast.*makeText*(Add\_Tutorial.this, R.string.*add\_im*, Toast.*LENGTH\_SHORT*).show();  else if (!fileIsSet)  Toast.*makeText*(Add\_Tutorial.this, R.string.*add\_file\_pl*, Toast.*LENGTH\_SHORT*).show();  else {  titleArt = TitleBody.getText().toString().trim();  final String keyDocument = uid + cc.getTimeInMillis();  StorageReference reference = FirebaseStorage.*getInstance*().  getReference("Images/" + uid + "/" + keyDocument);  final StorageReference photoArt = reference.child("thumbnail.png");  View vv = LayoutInflater.*from*(Add\_Tutorial.this).inflate(R.layout.*general\_layout\_image*,  null, false);  final AlertDialog dialog = new AlertDialog.Builder(Add\_Tutorial.this).setTitle(R.string.*uploding\_file*)  .setView(vv).create();  dialog.show();  if (typeof.equals("add")) {  uploadFromUri(file\_art,title);  }  }  }  }); }** | Lorsque l’utilisateur cliquer sur le bouton de publication.  Prendre des vérifications pour effectuer la publication :  Le titre ne doit pas être vide.  Ajouter une image.  Ajouter un fichier . |
| **private void uploadFromUri(Uri fileUri, String pdfname) {startService(new Intent(this, MyUploadService.class)  .putExtra(MyUploadService.*EXTRA\_FILE\_URI*, fileUri)  .putExtra(MyUploadService.*EXTRA\_IMAGE\_URI*, image\_art)  .putExtra("year", String.*valueOf*(Year))  .putExtra("title", pdfname)  .putExtra("module", Module)  .putExtra("pdf",false)  .putExtra("date",date)  .setAction(MyUploadService.*ACTION\_UPLOAD*));   Toast.*makeText*(this, R.string.*progress\_uploading*, Toast.*LENGTH\_LONG*).show();   Intent intent = new Intent(this, Profile\_Activity.class);  startActivity(intent);  finish(); }** | Enregistrer le fichier URI  Commencer MyUploadService pour télécharger le fichier,meme si cette activité est tué ou mise en arrière\_plan.  Selectionner le module. |

1. Afficher tous les tutoriels pour chaque année et pour chaque module :

**Class\_ShowAllTutorials.java :**

**public class ShowAllTutorials extends** AppCompatActivity {  
**public** DatabaseReference **reference** = **null**;  
**int what\_yearIs**;  
 String **moduleName**;  
 Query **query**;  
 FirebaseRecyclerAdapter<File\_item, ViewHolder>**adapter**;  
 FirebaseRecyclerOptions<File\_item>**options**;  
  
 @Override  
**protected void** onCreate(Bundle savedInstanceState) {  
**super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_show\_all\_tutorials***);  
 Bundle b = getIntent().getExtras();  
**if** (b.isEmpty())  
 onBackPressed();  
**else** {  
**what\_yearIs** = b.getInt(**"year"**);  
**moduleName** = b.getString(**"name"**);  
 }  
 updateView();  
**query** = FirebaseDatabase.*getInstance*()  
.getReference().child(**"Summaries"**).child(String.*valueOf*(**what\_yearIs**)).child(String.*valueOf*(**moduleName**));  
**options** = **new** FirebaseRecyclerOptions.Builder<File\_item>()  
 .setQuery(**query**, File\_item.**class**)  
 .build();  
 HandleTutos();

Récupérer les données àpartir de la base de données.

Quelle année et quel module.

\*Les classes secondaires utilisées :

**private void** updateView() {  
**if** (**what\_yearIs** == 1) {  
 ((TextView) findViewById(R.id.***title\_sub11***)).setText(R.string.***st\_year***);  
 ((TextView) findViewById(R.id.***title\_mod***)).setText(**moduleName**);  
 ((ImageView) findViewById(R.id.***icon\_year1***)).setImageDrawable(getDrawable(R.drawable.***ic\_one***));  
 } **else** {  
 ((ImageView) findViewById(R.id.***icon\_year1***)).setImageDrawable(getDrawable(R.drawable.***ic\_222***));  
 ((TextView) findViewById(R.id.***title\_sub11***)).setText(R.string.***st\_year***);  
 ((TextView) findViewById(R.id.***title\_mod***)).setText(**moduleName**) ;}}

Modifier la vue pour le module sélectionné.

**private void** SearchFor(String field) {

com.google.firebase.database.Query query1 = **query**.startAt(**"title"**, field)

.  
 endAt(**"title"**, field + **"\uf8ff"**);  
 FirebaseRecyclerOptions<File\_item> options1 = **new** FirebaseRecyclerOptions.Builder<File\_item>()  
 .setQuery(query1, File\_item.**class**)  
 .build();

**adapter**.updateOptions(options1);  
}

Recherche dans la liste des tutoriels publiés pour chaque module.

**public class** ViewHolder **extends** RecyclerView.ViewHolder {  
**private** MaterialCardView **root**;  
**private** TextView **txtTitle**;  
**private** ImageView **imagePoster**;  
**public** ViewHolder(**final** View itemView) {  
**super**(itemView);  
**txtTitle** = itemView.findViewById(R.id.***textGrid***);  
**imagePoster** = itemView.findViewById(R.id.***image\_game***);  
**root** =itemView.findViewById(R.id.***card\_n***); }

Occuper l’instianciation de la vue d’un seul index du recycleview

Récupérer tous les tutoriels publiés pour le module sélectionné à partir de la base de données.

**private void** HandleTutos() {  
  
 com.google.firebase.database.Query query = FirebaseDatabase.*getInstance*()  
 .getReference().child(**"Tutorials"**).child(String.*valueOf*(**what\_yearIs**)).limitToFirst(20).orderByChild(**"module"**).equalTo(**moduleName**);  
  
 FirebaseRecyclerOptions<File\_item> options = **new** FirebaseRecyclerOptions.Builder<File\_item>()  
 .setQuery(query, File\_item.**class**)  
 .build();  
  
  
  
**adapter** = **new** FirebaseRecyclerAdapter<File\_item, ViewHolder>(options) {  
 @NonNull  
 @Override  
**public** ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, **int** viewType) {  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.***courses\_file\_item***, parent, **false**);  
**return new** ViewHolder(view);  
 }  
  
 @Override  
**protected void** onBindViewHolder(@NonNull ViewHolder holder, **int** position, @NonNull File\_item model) {  
*//loader.hide();*holder.**txtTitle**.setText(model.getTitle());  
 holder.**root**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
**public void** onClick(View v) {  
 Bundle b = **new** Bundle();  
 File\_item item = **adapter**.getItem(position);  
 b.putString(**"date"**, item.getDate());  
 b.putString(**"title"**, item.getTitle());  
 b.putString(**"ref"**,**adapter**.getRef(position).toString());  
 b.putString(**"uid"**, item.getUid());  
 b.putString(**"link"**, item.getPostId());  
 b.putString(**"PostId"**, item.getPostId());  
 b.putString(**"image"**, item.getThumbnail());  
 Intent i = **new** Intent(getApplicationContext(), VideoPlayer.**class**);  
 i.putExtras(b);  
 startActivity(i);  
 }  
 });  
 holder.setImageFile(model.getThumbnail());  
  
 }  
  
 };  
 RecyclerView recyclerview = findViewById(R.id.***grid\_modules***);  
 recyclerview.setLayoutManager( **new** GridLayoutManager(ShowAllTutorials.**this**,2, GridLayoutManager.***VERTICAL***,**false**));  
 recyclerview.setItemViewCacheSize(50);  
 recyclerview.setAdapter(**adapter**); }

Adapter d’affichage la liste des tutoriels.

Un layoutInflater pour définir un fichier sélectionné parmi la liste.

ourses

Récupérer le tutoriel à partir d’adapter pour lire la vidéo.

Ajouter une image pour le tutoriel.

Un recyclerview pour afficher la liste des tutoriels.

**2-2/**Résumés :

On effectuonspresque les mêmes traitements pour les résumés, on utilisant les classes Add\_Summaries.java et ShowAllSummaries.java elles appliquent les mêmes opérations que Add\_Tutoriel.java et ShowAllTutorials.java respectivement.

***FIREBASE :***

Firebase vous offre des fonctionnalités telles que les analyses, les bases de données ,la messagrie et les rapports d’incidents pour que vous puissiez vous déplacer rapidement et vous concentrer sur vos utilisateurs, nous la utilisant pour manipuler notre application :

**1/Authentication :**

Firebase Authentification fournit des services de backend, des SDK faciles à utiliser et des bibliothèques d’interface utilisateur prêtes à l’emploi pour authentifier les utilisateurs auprès de votre application. Il prend en charge l’authentification à l’aide de mots de passe, de numéros de téléphone,de fournisseurs d’identité fédérés populaires tels que Google,Facebook,Twitter etc…

**Firebase Authentication avec Google :**

**Méthodes de connexion :**  il y a deux possibles méthodes pour se connecter :

**1-Avec email et mot de passe :**

**private void** signin() {  
 TextInputEditText SignUpMail = findViewById(R.id.***email\_field1***);  
 TextInputEditText SignUpPass = findViewById(R.id.***password\_field1***);  
 String email = SignUpMail.getText().toString();  
**final** String pass = SignUpPass.getText().toString();  
**if** (TextUtils.*isEmpty*(email)) {  
 SignUpMail.requestFocus();  
 SignUpMail.setError(getString(R.string.***empty\_email***));  
 } **else if** (!isValidEmail(email)) {  
 SignUpPass.requestFocus();  
 SignUpPass.setError(getString(R.string.***r\_u\_esist2***));  
 }  
**if** (TextUtils.*isEmpty*(pass)) {  
 SignUpPass.requestFocus();  
 SignUpPass.setError(getString(R.string.***empty\_password***));  
 } **else if** (!isValidPassword(pass)) {  
 SignUpPass.requestFocus();  
 SignUpPass.setError(getString(R.string.***short\_password***));  
 } **else if** (isValidEmail(email) && isValidPassword(pass)) {  
**firebaseAuth**.signInWithEmailAndPassword(email, pass)  
 .addOnCompleteListener(login.**this**, **new** OnCompleteListener<AuthResult>() {  
**public void** onComplete(@NonNull Task<AuthResult> task) {  
**if** (!task.isSuccessful()) {  
 Toast.*makeText*(login.**this**, R.string.***error*** + task.getException().getLocalizedMessage(),  
 Toast.***LENGTH\_LONG***).show();  
 } **else** {  
 Snackbar.*make*(**signin**, getString(R.string.***login\_successful***), Snackbar.***LENGTH\_LONG***).show();  
**final** FirebaseUser user = **firebaseAuth**.getCurrentUser();  
**if** (!user.isEmailVerified()) {  
 verify\_your\_email(user);  
 } **else** {  
 goToProfile();  
 }  
 }  
 }  
 });  
  
  
 }   
}

**2-Avec Google :**

**private void** signin\_google() {  
 GoogleSignInOptions gso = **new** GoogleSignInOptions.Builder(GoogleSignInOptions.***DEFAULT\_SIGN\_IN***)  
 .requestIdToken(getString(R.string.***default\_web\_client\_id***))  
 .requestEmail()  
 .build();  
 GoogleSignInClient mGoogleSignInClient = GoogleSignIn.*getClient*(**this**, gso);  
 Intent signInIntent = mGoogleSignInClient.getSignInIntent();  
 startActivityForResult(signInIntent, **RC\_SIGN\_IN**);  
  
  
}

**Vérification dés émails :**

\*Nous utilisons l’API email vérification pour vérifier l’existence, la validité et la qualité de toute adresse email :

https://emailverification.whoisxmlapi.com/api

**private void** verify\_your\_email(**final** FirebaseUser user) {  
  
**if** (user != **null** && user.getEmail().contains(**"@esi-sba.dz"**)) {  
 user.getUid();  
**if** (!user.isEmailVerified()) {  
**final** AlertDialog dialog = **new** AlertDialog.Builder(login.**this**).create();  
 dialog.setTitle(getString(R.string.***verify\_your\_email***));  
 dialog.setMessage(getString(R.string.***email\_sent***));  
  
 LayoutInflater inflater = LayoutInflater.*from*(login.**this**);  
**final** View login\_layout = inflater.inflate(R.layout.***general\_layout\_verify\_email***, **null**);  
  
**final** Button verify = login\_layout.findViewById(R.id.***verify***);  
 dialog.setView(login\_layout);  
 dialog.show();  
 verify.setOnClickListener(**new** View.OnClickListener() {  
@Override  
**public void** onClick(View v) {  
**firebaseAuth**.getCurrentUser().reload();  
**if** (!user.isEmailVerified()) {  
 Snackbar.*make*(verify, getString(R.string.***yremail\_not\_vrf***),  
 Snackbar.***LENGTH\_LONG***).show();  
 } **else** {  
dialog.dismiss();  
 goToProfile();  
 }  
 }  
 });  
  
 } **else** {  
 goToProfile();  
 }  
 }  
}

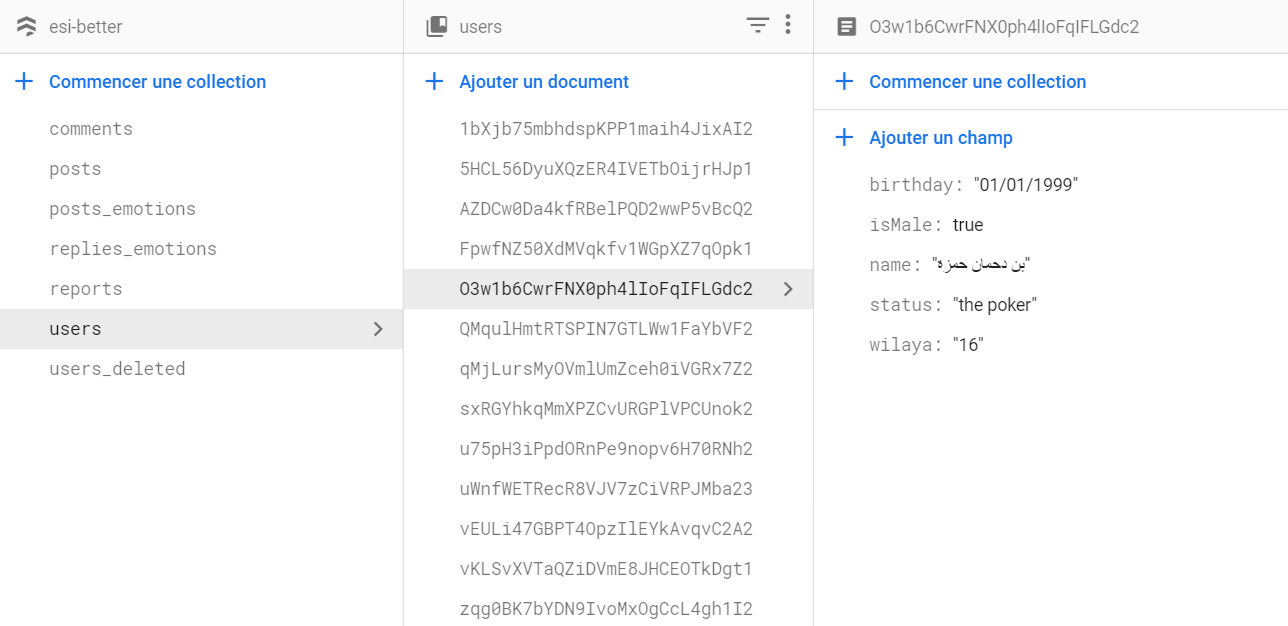
**liste des utilisateurs :**



**2/ Cloud Firestore :**

Cloud Firestore est une base de données NoSQL flexible et évolutive pour le développement mobile, Web et serveur de Firebase et de Google Cloud Platform.

En suivant le modèle de données NoSQL de Cloud Firestore, vous stockez les données dans des documents contenant des champs mappés à des valeurs. Ces documents sont stockés dans des collections, qui sont des conteneurs pour vos documents que vous pouvez utiliser pour organiser vos données et créer des requêtes. Les documents prennent en charge de nombreux [types de données](https://firebase.google.com/docs/firestore/manage-data/data-types) , des chaînes simples et des nombres aux objets complexes et imbriqués. Vous pouvez également créer des sous-collections dans des documents et créer des structures de données hiérarchiques qui évoluent à mesure que votre base de données se développe. Le [modèle de données](https://firebase.google.com/docs/firestore/data-model) Cloud Firestore prend en charge la structure de données la mieux adaptée à votre application.

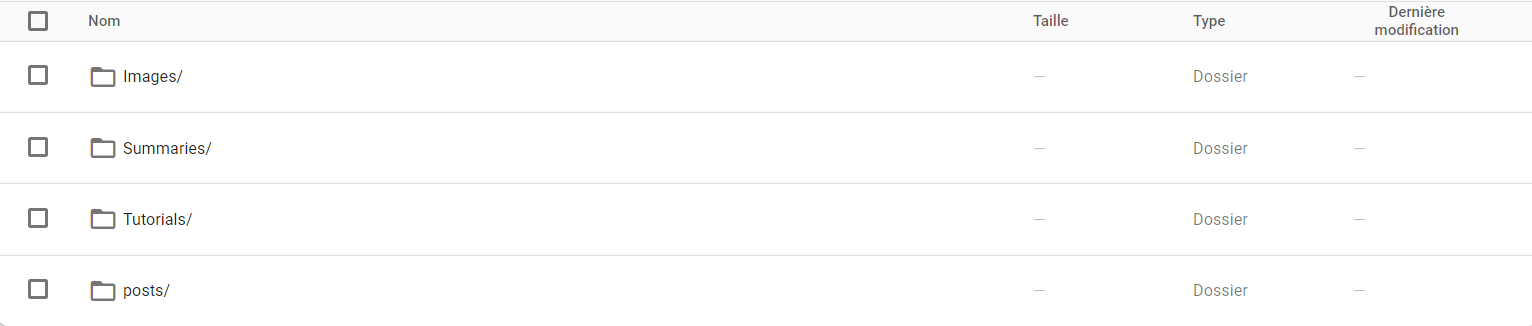


Collections documents champs

**3/Cloud Storage :**

Cloud Storage for Firebase est un service de stockage d'objets puissant, simple et économique conçu pour l'échelle de Google.

Nous utilisons Cloud storage car nous avons besoin de stocker et de diffuser du contenu généré par l'utilisateur, tel que lesimages et tutoriels et les résumés :



**4/Realtime Database :**

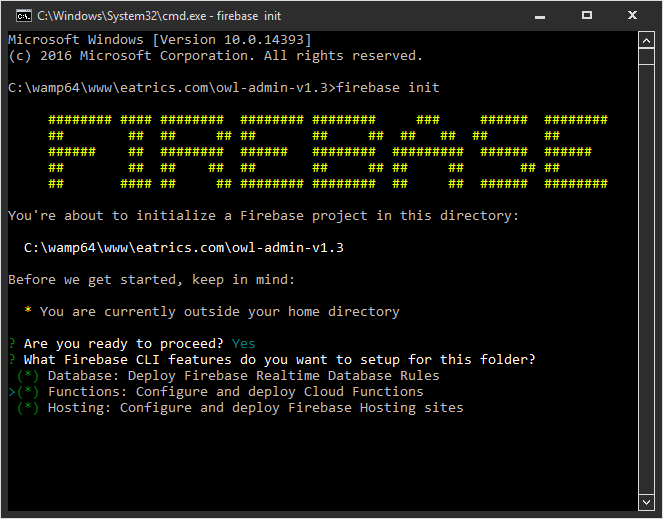
La base de données Firebase Realtime est une base de données hébergée dans le cloud. Les données sont stockées au format JSON et synchronisées en temps réel avec chaque client connecté. Lorsque vous créez des applications multiplateformes avec les SDK iOS, Android et JavaScriptde Firebase, tous vos clients partagent une instance de base de données en temps réel et reçoivent automatiquement des mises à jour avec les données les plus récentes :



**4/Firebase Cloud Functions :**

Cloud Functions for Firebase est un framework sans serveur qui vous permet d'exécuter automatiquement du code backend en réponse aux événements déclenchés  ;Triggers ; par les fonctionnalités Firebase et les requêtes HTTPS. Votre code JavaScript ou TypeScript est stocké dans le cloud de Google et s'exécute dans un environnement géré. Il n'est pas nécessaire de gérer et de mettre à l'échelle vos propres serveurs

On a choisi la language Node js por la déploiments des Fonctionements . et on a utilise Firebase CLI comme environement .



**exports.sendLikeNotification = functions.firestore.document("posts/{post\_id}").onWrite(async (change,context)=> {**

**const post\_id = context.params.post\_id;**

**let tokenid;**

**let newPost ;**

**const admin\_id = change.after.data().uid;**

**const tokenRef = admin.database().ref('users/'+admin\_id);**

**const likes\_num = change.after.data().likes;**

**const dislikes\_num = change.after.data().dislikes;**

**const body = change.after.data().body;**

**const date = change.after.data().date;**

**newPost = likes\_num === 0 && dislikes\_num === 0 ;**

**if (newPost)**

**return null ;**

**const post\_name = change.after.data().title;**

**const image = change.after.data().image;**

**// The array containing all the user's tokens.**

**await tokenRef.once("value").then(function(snapshot) {**

**return tokenid = snapshot.val();**

**});**

**const payload = {**

**data:{**

**post\_id:post\_id,**

**body:body,**

**title: post\_name,**

**date: date,**

**reply: "",**

**uid: admin\_id,**

**image: image,**

**name: "",**

**number : "",**

**comment: "",**

**likes:likes\_num.toString(),**

**dislikes:dislikes\_num.toString(),**

**type:"like"**

**}**

**};**

**console.log("liked");**

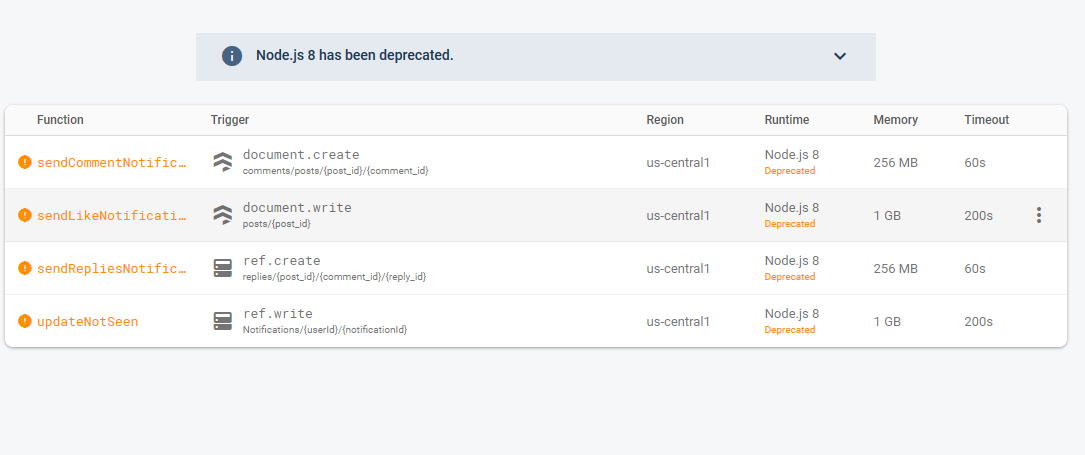
**const time = Date.now();**

**const userNotification = admin.database().ref('/Notifications/'+admin\_id+'/'+post\_id);**

**await userNotification.set({seen: false, post\_id: post\_id, time: time, title: post\_name, likes:likes\_num, name: "", number : "", comment: "",reply: "", date: date, image: image, body: body, dislikes:dislikes\_num, type:"like" });**

**return admin.messaging().sendToDevice(tokenid, payload);**

});

******

***LES PERMISSIONS :***

\*Accès à Internet.

\*Accès au Stockage.

\*Accès pour envoyer des notifications.

***Aperçu sur notre application :***

On vous présente notre application ESI better

