|  |
| --- |
| **DEVIS MINISTÉRIEL** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Code de la compétence** | 00SS |  |  | **Contexte de réalisation** |
| **Énoncé de la compétence** | | |  | -Pour différentes plateformes cibles : tablettes, téléphones intelligents, ordinateurs de bureau, etc.  -Pour de nouvelles applications et des applications à modifier.  -À partir des documents de conception.  -À l’aide d’un compilateur conçu pour la plateforme cible, d’un compilateur croisé ou d’un interpréteur.  -À l’aide d’un émulateur sur la plateforme hôte.  -À l’aide d’images, de sons et de vidéos.  -À l’aide de procédures de suivi des problèmes et de gestion des versions |
| Effectuer le développement d’applications natives avec base de données | | |
| **Précisions ministérielles (s’il y a lieu)** | | |  |
|  | | |

|  |
| --- |
| **INTERPRÉTATION LOCALE** |

|  |  |
| --- | --- |
| **PLACE DE LA COMPÉTENCE DANS LE PROGRAMME** | |
| **Axe du programme**  **(*En fonction du profil de sortie)*** | Programmation native |
| **Lien(s) avec les autres compétences**  **(*En lien avec le logigramme de compétences)*** | Cette compétence est la suite de la compétence 00SR.  L’étudiante et l’étudiant apprendront à concevoir des applications natives utilisant une base de données. Ces applications pourront être sur différents types de plateformes (ex : ordinateur, tablette, téléphone). |
| **Précisions nécessaires à la compréhension de la compétence**  **(*En fonction du ou des cours associés)*** | Cette compétence est répartie sur 2 cours afin de séparer divers aspects de la programmation native avec base de données.  Dans le cours 420-4N1-DM développement d’application native III, les notions de base de données apprises dans le cours 420-3N3-DM base de données seront intégrées au développement d’applications natives plus complexes. Des notions plus avancées de base de données seront aussi démontrées.  Dans le cours 420-6N1-DM développement d’applications native IV, toutes les notions apprises dans l’axe de programmation native seront mises en œuvre dans un projet conçu et développé entièrement par l’étudiante ou l’étudiant. |
| **Terminologie et règles utilisées dans les objets d’apprentissage et dans les balises de contenu.** | **Compilateur croisé :** Compilateur capable de produire un code exécutable à destination d'architectures différentes de celle sur laquelle la compilation est réalisée (ex : développement sur ordinateur de bureau et compilation pour un téléphone intelligent).  **Plateforme de compilation** : La plateforme sur laquelle la compilation est faite (habituellement un ordinateur de bureau).  **Plateforme d’exécution** : Dans le cas d’une compilation non-croisée, la plateforme de compilation et d’exécution sont les mêmes.  **Plateforme d’exécution croisée** : Dans le cas d’une compilation croisée, l’exécution se fait sur une plateforme autre que la plateforme de compilation (ex : téléphone, tablette, émulateur).  **Émulateur** : Logiciel permettant de simuler le fonctionnement d’une autre plateforme (ex : simulation d’un téléphone sur un ordinateur)  **Cadriciel de BD** : librairie permettant d’isoler la base de données de l’application, ou permettant de faire des requêtes à la bd sans utiliser un langage propre aux bd tel que sql (ex : un ORM)  **Usage du singulier** : dans ce document, le singulier est utilisé afin de ne pas alourdir le document, par exemple : utilisation d’un item (ex : langage). Si l’enseignant estime que plusieurs items devraient être enseignés afin de couvrir les connaissances nécessaires à la réalisation de la compétence, il est libre d’en utiliser plusieurs.  **Usage du pluriel** : Si le pluriel est utilisé dans la description, l’enseignant doit alors montrer plusieurs items.  **Utilisation de principes orientés objets** : Dans ces cours, l’approche orientée objet peut être utilisée, mais n’est pas obligatoire. Il est donc recommandé, lorsque possible, d’utiliser des techniques orientées objet telles que langage supportant les objets, des modèles objets, etc.  **Document de conception** : Tous documents permettant de comprendre le projet à développer. Le terme « conception » ne doit pas être pris dans le sens de « analyse suivie de conception ». Les documents suivants sont des exemples de documents de conception : l’énoncé du travail pratique; un diagramme de l’architecture demandée (ex : réseau, bd, objets); la demande du client; un dossier de conception complet; etc. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **INFORMATIONS GÉNÉRALES SUR LES COURS ASSOCIÉS À LA COMPÉTENCE** | | | | | | | | |
| **Code du cours** | | **Titre du cours** | **Pondération** | **Heures-contact** | **Heures attribuées à la compétence dans le cours** | **Trimestre** | **Préalable(s)** | | | |
|  | |  |  |  |  |  | **Ce cours a comme préalable** | **Ce cours est préalable** | | |
| 1 | 420-4N1-DM | Développement d’applications natives III | 2-2-1 | 60 | 60 | 4 | 420-3N1-DM Programmation Orientée Objet II,  420-3N2-DM Programmation Native II,  420-3N3-DM Base de données | 420-5W1-DM Projet Web I,  420-6N1-DM Développement d’applications natives IV | |
| 2 | 420-6N1-DM | Développement d’applications natives IV | 1-3-4 | 60 | 60 | 6 | 420-4N1-DM Développement d’applications natives III | Aucun cours | | |
|  |  |  |  | **Total 120** | **Total 120** |  |  |  | | |

**Cours 1** : Développement d’applications natives III

|  |  |
| --- | --- |
| **PLACE DU COURS DANS LE PROGRAMME** | |
| **Lien(s) avec les autres cours** | Ce cours a comme préalable relatif 420-3N1-DM Programmation Orientée Objet II, 420-3N2-DM Programmation Native II et 420-3N3-DM Base de données.  Il est préalable à 420-5W1-DM Développement d’applications web I, et 420-6N1-DM Développement d’applications natives IV |
| **(*En lien avec le logigramme de cours)*** |
| **Rôle du cours dans le programme** | Ce cours se donne à la quatrième session.  Il sert à introduire la programmation native utilisant une base de données |
| ***(Initiation, application, intégration)*** |
| **Lien(s) avec le projet éducatif** | L’accessibilité aux études et aux savoirs ainsi que la réussite sont les deux valeurs fondamentales du projet éducatif du Cégep de Drummondville auxquelles les enseignantes et les enseignants du département des techniques de l’informatique souscrivent entièrement. Cette adhésion au projet éducatif se manifeste notamment par la mise en œuvre de moyens d’enseignement qui favorisent la transmission des connaissances, le développement de compétences durables et transférables et la reconnaissance que l’étudiante ou l’étudiant est le principal agent de son succès par son engagement personnel et entier dans son projet de formation. Soucieux de fournir un enseignement de qualité, les enseignantes et les enseignants du département des techniques de l’informatique se sont donc donné des règles qui permettent l’apprentissage dans un climat propice aux études et qui mettent de l’avant des valeurs comme l’honnêteté intellectuelle, l’éthique de travail, l’expression de la pensée dans une langue correcte, la justice en matière d’évaluation et l’accessibilité à des ressources professionnelles accueillantes et compétentes. |

**Cours 2** : Développement d’applications natives IV

|  |  |
| --- | --- |
| **PLACE DU COURS DANS LE PROGRAMME** | |
| **Lien(s) avec les autres cours** | Ce cours a comme préalable le cours 420-4N1 Développement d’applications natives III.  Il n’est préalable à aucun cours. |
| **(*En lien avec le logigramme de cours)*** |
| **Rôle du cours dans le programme** | Ce cours se donne à la sixième session.  Il sert à approfondir la programmation sur plateforme native utilisant une base de données et introduire la partie « cible » du développement d’application native avec base de données.  Le projet développé dans ce cours devrait suivre les principes du logiciel libre |
| ***(Initiation, application, intégration)*** |
| **Lien(s) avec le projet éducatif** | L’accessibilité aux études et aux savoirs ainsi que la réussite sont les deux valeurs fondamentales du projet éducatif du Cégep de Drummondville auxquelles les enseignantes et les enseignants du département des techniques de l’informatique souscrivent entièrement. Cette adhésion au projet éducatif se manifeste notamment par la mise en œuvre de moyens d’enseignement qui favorisent la transmission des connaissances, le développement de compétences durables et transférables et la reconnaissance que l’étudiante ou l’étudiant est le principal agent de son succès par son engagement personnel et entier dans son projet de formation. Soucieux de fournir un enseignement de qualité, les enseignantes et les enseignants du département des techniques de l’informatique se sont donc donné des règles qui permettent l’apprentissage dans un climat propice aux études et qui mettent de l’avant des valeurs comme l’honnêteté intellectuelle, l’éthique de travail, l’expression de la pensée dans une langue correcte, la justice en matière d’évaluation et l’accessibilité à des ressources professionnelles accueillantes et compétentes. |

| **Titre du cours :** | 1 | Développement d’applications natives III | **Code du cours :** | 420-4N1-DM | **Trimestre :** | 4 |
| --- | --- | --- | --- | --- | --- | --- |

**Règles générales pour les balises de l’évaluation**

À moins de précisions, les règles définies à la section 3.1.3 de la PDÉA s’appliquent.

Pour ce cours, les règles suivantes sont recommandées :

* Une évaluation formative ou sommative devrait avoir été faite et corrigée avant la date d’abandon;
* L’évaluation finale devrait avoir une pondération minimale de 35%;
* L’ensemble des autres évaluations doit avoir une pondération totale minimale de 30% et maximale de 65%.

Types d’évaluations recommandés pour l’évaluation finale:

* ~~Devoir;~~
* Projet;
* ~~Examen pratique;~~
* ~~Examen théorique;~~
* ~~Examen théorique et pratique;~~
* Rapport.

Types d’évaluations recommandés pour les autres évaluations:

* Devoir;
* Projet;
* Examen pratique;
* Examen théorique;
* Examen théorique et pratique;
* Rapport.

Si un pourcentage est indiqué dans la colonne balise de l’évaluation d’un élément de compétence, celui-ci indique l’importance relative que cet élément **devrait** avoir dans le cours. Ce pourcentage est donné **à titre indicatif**. Il est possible que le total pour le cours ne donne pas 100%.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 1. Analyser le projet de développement de l’application. | 1.1 Analyse juste des documents de conception. | -Interpréter les documents de conception.  -Appliquer des techniques simples d’analyse de documents.  -Identifier les composants du système à programmer (ex : interface usager, base de données).  -Organiser les tâches nécessaires à la réalisation du projet. | -La conversion des besoins exprimés dans un document de conception en spécification d’application native à l’aide de techniques simples d’analyse.  -La lecture d’un document de conception.  -La priorisation des composants à réaliser.  -La création d’un calendrier des tâches à effectuer. | 5% | -Le terme « Analyser » doit être pris dans le sens de « comprendre » et non pas de générer un document d’analyse.  -Le projet peut être imposé par l’enseignant ou l’enseignante, ou être choisi par l’étudiante ou l’étudiant. |
|  | 1.2 Détermination correcte des tâches à effectuer |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 2. Préparer l’environnement de développement informatique | 2.1 Installation correcte des logiciels et des bibliothèques sur la plateforme hôte. | -Installer un environnement permettant le développement d’application native à l’aide d’un langage interprété ou compilé.  -Installer ~~l~~es outils de débogage sur la plateforme d’exécution.  -Tester l’environnement installé | -L’installation de l’environnement de développement.  -L’installation et l’utilisation d’un débogueur.  -La vérification de l’environnement (ex : compilation simple, exécution, débogage) | 5% | -Le cours 420-2N3-DM Outils de développement couvre en détails l’installation d’un environnement de travail. L’emphase devra donc être mise sur les particularités de l’environnement spécifique à ce cours. |
|  | 2.2 Configuration appropriée de la plateforme cible |  |  | -Vu dans 420-6N1-DM  -Ce cours ne couvre que la partie hôte. |
|  | 2.3 Configuration appropriée du système de gestion de versions. | -Configurer un système de gestion de version dans un environnement de développement natif.  -Tester le système de gestion de versions | -L’intégration du projet dans un système de version.  -L’intégration du système de version dans le système de développement natif.  -La vérification du bon fonctionnement du système de gestion de versions (ex : faire une soumission) | -La configuration d’un système de gestion de versions est couvert dans le cours 420-2N3 Outils de développement. Seules les particularités spécifiques à l’environnement utilisé dans ce cours doivent être couvertes.  -Le développement peut se faire seul ou en équipe. |
|  | 2.4 Importation correcte du code source | -Utiliser un outil de gestion de version afin d’importer du code.  -Utiliser les bonnes pratiques d’importation de code source  -Tester l’importation | -Les bonnes pratiques d’importation (ex : git clone)  -Les techniques de base pour vérifier si le code est bien importé (ex : comparer avec le dépôt, checksum) | -Des exemples de code peuvent être fournis par l’enseignant dans un dépôt. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 3. Préparer la ou les bases de données. | 3.1 Création ou adaptation correctes de la base de données locale ou de la base de données distante. | -Utiliser un outil de création et de modification de base de données locale ou distante propre à l’environnement sélectionné.  -Créer une base de données simple, locale ou distante à partir des documents de conception. | -L’utilisation d’automatisation pour la création de la bd (ex : script, migration)  -La création ou l’adaptation de la bd à partir de l’analyse des documents de conception.  -Les techniques et outils de création d’une bd. | 20% | -Pour clarification, un exemple de bd locale : sqlite; une bd distante : sql server, mysql. |
|  | 3.2 Insertion correcte des données initiales ou de données de tests. | -Utiliser un script simple de création de données (ex : seeder).  -Utiliser un outil de création de masse (ex : faker).  -Choisir les données initiales. | -Les outils de création de données (ex : seeder, faker pour données simple).  -Le choix des données initiales (ex : compte admin, listes prédéfinies)  -Le choix des données de tests. |  |
|  | 3.3 Respect du modèle de données. | -Identifier les différences entre le modèle et la bd réalisée.  -Corriger le modèle ou la bd afin d’assurer la cohérence. | -La traduction du langage de représentation du modèle de données en langage de représentation de la bd.  -La correction d’erreurs dans le modèle de données  -La correction d’erreurs dans la structure de la bd (ex : migrations supplémentaires) |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 4. Générer ou programmer l’interface graphique. | 4.1 Choix et utilisation appropriés des éléments graphiques pour l’affichage et la saisie. | ~~-~~Choisir les éléments graphiques pour la plateforme d’exécution.  -Comparer les caractéristiques des éléments graphiques.  -Utiliser les éléments graphiques correctement. | -Les différents éléments graphiques disponibles sur la plateforme d’exécution et leurs caractéristiques (ex : bouton, grille, checkbox, etc.)  -Le rôle standard de chaque élément graphique |  |  |
|  | 4.2 Intégration correcte des images. | -Intégrer des images dans l’application pour la plateforme d’exécution  -Appliquer les bonnes pratiques pour l’intégration des images. | -Les techniques pour intégrer des images dans une application  -Les bonnes pratiques pour l’intégration d’une image. |  |  |
|  | 4.3 Adaptation de l’interface en fonction du format d’affichage et de la résolution. | -Appliquer les concepts d’affichage adaptatif pour la plateforme d’exécution. | -Les concepts d’affichage adaptatif. |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| **5.** Programmer la logique applicative. | 5.1 Programmation ou intégration correctes de mécanismes d’authentification et d’autorisation. | -Choisir les mécanismes d’authentification et d’autorisation.  -Développer les mécanismes d’authentification et d’autorisation en utilisant l’environnement de développement. | -Les mécanismes d’authentification (ex : mot de passe, biométrique).  -Les mécanismes d’autorisation (ex : par sous-système, par rôle).  -L’intégration de ces mécanismes dans l’application. | 40% |  |
|  | 5.2 Programmation correcte des interactions entre l’interface graphique et l’utilisatrice ou l’utilisateur. | Cet élément de compétence, ainsi que ses objets d’apprentissage et ses balises de contenu sera enseigné dans le cours 420-6N1-DM | | | |
|  | 5.3 Choix approprié des clauses, des opérateurs, des commandes ou des paramètres dans les requêtes à la base de données. | -Utiliser l’environnement de développement pour construire les requêtes à la base de données. | -Les commandes pour accéder à la base de données.  -L’utilisation d’un cadriciel de BD (ex : ORM) |  | -Les techniques de base pour accéder à une bd sont enseignées dans 420-3N3-DM Base de données. Des techniques plus avancées, tel que l’utilisation d’ORM, doivent être utilisées dans ce cours. |
|  | 5.4 Manipulation correcte des données de la base de données. | -Choisir la méthode appropriée pour accéder aux données dans la bd.  -Formuler les requêtes à la bd selon le langage applicatif et l’environnement de programmation utilisé. | -Les techniques d’accès aux données de la bd (ex : transactions, lazy loading vs eager loading).  -L’écriture des requêtes selon l’environnement de programmation (ex : sql vs nosql). |  |  |
|  | 5.5 Programmation correcte de la synchronisation des données. | Cet élément de compétence, ainsi que ses objets d’apprentissage et ses balises de contenu sera enseigné dans le cours 420-6N1-DM | | | |
|  | 5.6 Utilisation appropriée des services d’échange de données. | -Utiliser une technique d’échange de données.  -Tester le service d’échange de données | -Une technique d’échange de données (ex : API via http)  -Les techniques pour tester un service d’échange de données (ex : tests automatisés). |  |  |
|  | 5.7 Application correcte des techniques d’internationalisation. | -Utiliser des techniques d’internationalisation. | -Les techniques d’internationalisation en programmation native. |  |  |
|  | 5.8 Application rigoureuse des techniques de programmation sécurisée | -Utiliser les bonnes pratiques en matière de programmation sécurisée.  -Utilisation appropriée des stratégies de sécurisation des données entrées par l’utilisatrice et l’utilisateur sur la plateforme d’exécution.  - Utilisation appropriée des techniques de contrôle des erreurs et des exceptions sur la plateforme d’exécution. | -les bonnes pratiques en matière de programmation sécurisée (ex : validation des données, valeur limite, valeur erronée, la gestion des erreurs à l’aide d’exceptions, l’encryption de données) |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 6. Contrôler la qualité de l’application. | 6.1 Application rigoureuse des plans de tests sur l’émulateur et sur la plateforme cible. | -Concevoir un plan de test pour la plateforme d’exécution  -Concevoir un plan de test pour la plateforme d’exécution croisée.  -Appliquer les plans de tests. | -Décrire le plan/jeu de tests  -Exécuter le plan/jeu de tests  -Les outils d’exécution de jeu de tests sur la plateforme d’exécution. | 20% | -La partie émulateur est couverte dans 420-6N1-DM |
|  | 6.2 Revues de code et de sécurité rigoureuses. | -Appliquer les standards de programmation native.  -Critiquer la qualité de son code et du code d’un pair. | -Les standards de programmation des langages utilisés~~.~~  -Les techniques de revue de code. |  |
|  | 6.3 Pertinence des correctifs | -Noter l’information nécessaire à la reproduction de l’erreur.  -Démontrer la pertinence du correctif. | -La documentation de l’erreur et du correctif.  -La reproduction de l’erreur et la vérification de la correction (ex : test de non-régression). |  |
|  | 6.4 Respect des procédures de suivi des problèmes et de gestion des versions. | ~~-~~Utiliser un système de gestion de versions dans un environnement de développement natif. | -Les particularités de l’utilisation du système de gestion de version dans l’environnement de développement utilisé. | -Le système de gestion de version est couvert dans le cours 420-2N3-DM Outils de développement. compétence 00Q6, élément 7.  -La partie relative au « suivi des problèmes » est couverte dans 420-3N1-DM Développement d’applications natives II.  -Ce critères est aussi vu dans la compétence 00ST. Il faudra donc coordonner avec le cours porteur |
|  | 6.5 Respect des documents de conception. | -Évaluer la qualité de l’application. | -Les techniques d’assurance qualité.  -Les essais de système et/ou d’acceptation. | -Les documents de conception sont produits par l’enseignant. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 7. Participer à la mise en service de l’application. | 7.1 Préparation appropriée de l’application en vue de son déploiement ou de son installation. | Cet élément de compétence, ainsi que ses objets d’apprentissage et ses balises de contenu sera enseigné dans le cours 420-6N1-DM | | | |
|  | 7.2 Déploiement ou installation corrects de l’application. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 8. Rédiger la documentation. | 8.1 Détermination correcte de l’information à rédiger. | -Utiliser les bonnes pratiques pour l’écriture de la documentation  -Écrire de la documentation pour l’usager.  -Écrire la documentation du code. | -Les bonnes pratiques pour la rédaction de la documentation.  -Les outils d’écriture de documentation.  -L’écriture de la documentation pour l’usager (ex : aide en ligne, manuel d’usager).  -L’écriture de la documentation du code (ex : commentaire de fonction, documentation d’API) | 5% |  |
|  | 8.2 Notation claire du travail effectué. |  |

|  |  |
| --- | --- |
| **Médiagraphie indicative :** |  |
|  |

| **Titre du cours :** | 2 | Développement d’applications natives IV | **Code du cours :** | 420-6N1-DM | **Trimestre :** | 6 |
| --- | --- | --- | --- | --- | --- | --- |

**Règles générales pour les balises de l’évaluation**

À moins de précisions, les règles définies à la section 3.1.3 de la PDÉA s’appliquent.

Pour ce cours, les règles suivantes sont recommandées :

* Une évaluation formative ou sommative devrait avoir été faite et corrigée avant la date d’abandon;
* L’évaluation finale devrait avoir une pondération minimale de 35%;
* L’ensemble des autres évaluations doit avoir une pondération totale minimale de 30% et maximale de 65%.

Types d’évaluations recommandés pour l’évaluation finale:

* Devoir;
* Projet;
* Examen pratique;
* Examen théorique;
* Examen théorique et pratique;
* Rapport.

Types d’évaluations recommandés pour les autres évaluations:

* Devoir;
* Projet;
* Examen pratique;
* Examen théorique;
* Examen théorique et pratique;
* Rapport.

Si un pourcentage est indiqué dans la colonne balise de l’évaluation d’un élément de compétence, celui-ci indique l’importance relative que cet élément **devrait** avoir dans le cours. Ce pourcentage est donné **à titre indicatif**. Il est possible que le total pour le cours ne donne pas 100%.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 1. Analyser le projet de développement de l’application. | 1.1 Analyse juste des documents de conception. | ~~-~~Modéliser les documents de conception pour une application natives.  -Planifier l’organisation des tâches  -Organiser les tâches | -La conversion des besoins exprimés dans un document de conception en spécifications d’applications natives.  -Les techniques simples de planification des tâches. (ex : agenda, kanban, gantt) | 10% | Le terme « Analyser » doit être pris dans le sens de « comprendre » et non pas de générer un document d’analyse  -Les documents de conception seront fournis par l’étudiant ou l’étudiante.  -Les documents de conception sont tous les documents permettant de comprendre le projet. L’étudiant ou l’étudiante devra donc fournir une explication claire du projet qu’il ou elle désire réaliser.  -. |
|  | 1.2 Détermination correcte des tâches à effectuer |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 2. Préparer l’environnement de développement informatique | 2.1 Installation correcte des logiciels et des bibliothèques sur la plateforme hôte. | ~~-~~Installer un environnement permettant le développement d’application native.  ~~-~~Installer un émulateur  -Installer les outils de débogage.  -Tester l’environnement installé | -L’installation de l’environnement de développement.  -L’installation d’un émulateur.  -L’installation et l’utilisation d’un débogueur sur la plateforme d’exécution et sur l’émulateur.  -La vérification de l’environnement (ex : compilation simple, exécution, débogage) | 10% |  |
|  | 2.2 Configuration appropriée de la plateforme cible | -Configurer la plateforme cible afin de pouvoir accueillir le code généré sur la plateforme hôte. | - La préparation de la plateforme cible choisie pour le cours.  -Les étapes de transfert de l’application de la plateforme hôte à la cible. |  |
|  | 2.3 Configuration appropriée du système de gestion de versions. | ~~-~~Configurer un système de gestion de version dans un environnement multi-usagers | -L’intégration du projet dans un système de version.  -L’intégration du système de version dans le système de développement natif. |  |
|  | 2.4 Importation correcte du code source | -Utiliser un outil de gestion de version afin d’importer du code dans un environnement multi-usagers.  -Appliquer les bonnes pratiques d’importation de codes. | -l’utilisation d’un outil de gestion de versions dans un environnement multi-usagers.  -Les bonnes pratiques d’importation de code en mode multi-usagers. (ex: résolution de conflits, pull request). |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 3. Préparer la ou les bases de données. | 3.1 Création ou adaptation correctes de la base de données locale ou de la base de données distante. | -Utiliser un outil de création et de modification de base de données locale ou distante propre à l’environnement sélectionné.  -Créer une base de données plus complexe, locale ou distante à partir des documents de conception. | -L’utilisation d’automatisation pour la création de la bd (ex : script, migration)  -La création ou l’adaptation de la bd à partir de l’analyse des documents de conception.  -Les techniques et outils de création d’une bd. |  |  |
|  | 3.2 Insertion correcte des données initiales ou de données de tests. | -Utiliser un script pour créer des données plus complexes (ex : seeder).  -Utiliser un outil de création de masse pour des données plus complexes (ex : faker).  -Choisir les données initiales. | -Les outils de création de données plus complexe (ex : seeder associant à des données pré-existantes, faker pour des types de données complexes).  -Le choix des données de tests plus complexes. |  |  |
|  | 3.3 Respect du modèle de données. | Cet élément de compétence, ainsi que ses objets d’apprentissage et ses balises de contenu sera enseigné dans le cours 420-4N1-DM | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 4. Générer ou programmer l’interface graphique. | 4.1 Choix et utilisation appropriés des éléments graphiques pour l’affichage et la saisie. | Cet élément de compétence, ainsi que ses objets d’apprentissage et ses balises de contenu sera enseigné dans le cours 420-4N1-DM | | | |
|  | 4.2 Intégration correcte des images. |
|  | 4.3 Adaptation de l’interface en fonction du format d’affichage et de la résolution. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| **5.** Programmer la logique applicative. | 5.1 Programmation ou intégration correctes de mécanismes d’authentification et d’autorisation. | -Choisir les mécanismes d’authentification et d’autorisation.  -Développer les mécanismes d’authentification et d’autorisation. | -Les mécanismes d’authentification (ex : mot de passe, biométrique).  -Les mécanismes d’autorisation (ex : par sous-système, par rôle).  -L’intégration de ces mécanismes dans l’application. |  |  |
|  | 5.2 Programmation correcte des interactions entre l’interface graphique et l’utilisatrice ou l’utilisateur. | -Choisir le modèle d’interaction.  -Développer les interactions à l’aide de l’environnement.  -Utiliser les bonnes pratiques de développement d’interactions. | -Les modèles d’interaction (ex : graphique vs console, Windows vs Mac vs Linux).  -Les bonnes pratiques de développement d’interactions. |  |  |
|  | 5.3 Choix approprié des clauses, des opérateurs, des commandes ou des paramètres dans les requêtes à la base de données. | -Utiliser l’environnement pour construire les requêtes à la base de données. | -Les commandes pour accéder à la base de données.  -L’utilisation d’un cadriciel de BD (ex : ORM) |  |  |
|  | 5.4 Manipulation correcte des données de la base de données. | -Choisir la méthode appropriée pour accéder aux données dans la bd.  -Formuler les requêtes à la bd selon le langage applicatif et l’environnement de programmation utilisé. | -Les techniques d’accès aux données de la bd (ex : transactions, lazy loading vs eager loading).  -L’écriture des requêtes selon l’environnement de programmation (ex : sql vs nosql). |  |  |
|  | 5.5 Programmation correcte de la synchronisation des données. | -Utiliser des techniques de synchronisation des données  -Tester la synchronisation | -Les techniques de synchronisation des données (ex : détection de maj, résolution de conflits).  -Les techniques de test de la synchronisation. |  | -Exemple minimal d’utilisation de synchronisation : plusieurs clients utilisent la même BD en même temps. |
|  | 5.6 Utilisation appropriée des services d’échange de données. | -Utiliser une technique d’échange de données.  -Tester le service d’échange de données | -Une technique d’échange de données (ex : API via http)  -Les techniques pour tester un service d’échange de données (ex : tests automatisés). |  |  |
|  | 5.7 Application correcte des techniques d’internationalisation. | -Utiliser des techniques d’internationalisation sur la plateforme d’exécution. | -Les techniques d’internationalisation en programmation native pour la plateforme d’exécution. | 5% |  |
|  | 5.8 Application rigoureuse des techniques de programmation sécurisée | -Utiliser les bonnes pratiques en matière de programmation sécurisée.  -Utilisation appropriée des stratégies de sécurisation des données entrées par l’utilisatrice et l’utilisateur sur la plateforme d’exécution.  - Utilisation appropriée des techniques de contrôle des erreurs et des exceptions sur la plateforme d’exécution. | -les bonnes pratiques en matière de programmation sécurisée (ex : validation des données, valeur limite, valeur erronée, la gestion des erreurs à l’aide d’exceptions, l’encryption de données) | 5% |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 6. Contrôler la qualité de l’application. | 6.1 Application rigoureuse des plans de tests sur l’émulateur et sur la plateforme cible. | -Concevoir un plan de test pour la plateforme d’exécution | -Décrire le plan/jeu de tests  -Exécuter le plan/jeu de tests  -Les outils d’exécution de jeu de tests sur la plateforme d’exécution. | 5% |  |
|  | 6.2 Revues de code et de sécurité rigoureuses. | -Appliquer les standards de programmation native.  -Critiquer la qualité de son code et du code d’un pair. | -Les standards de programmation des langages utilisés~~.~~  -Les techniques de revue de code. |  |
|  | 6.3 Pertinence des correctifs | -Noter l’information nécessaire à la reproduction de l’erreur.  -Démontrer la pertinence du correctif. | -La documentation de l’erreur et du correctif.  -La reproduction de l’erreur et la vérification de la correction (ex : test de non-régression). |  |
|  | 6.4 Respect des procédures de suivi des problèmes et de gestion des versions. | ~~-~~Utiliser un système de gestion de versions dans un environnement de développement natif. | -Les particularités de l’utilisation du système de gestion de version dans l’environnement de développement utilisé. | -La configuration et l’utilisation d’un système de gestion de version sont couvertes dans le cours 420-2N3-DM Outils de développement, compétence 00Q6, élément 7.  -La partie relative au « suivi des problèmes » est couverte dans 420-3N2-DM Développement d’applications natives II.  -Ce critères est aussi vu dans la compétence 00ST. Il faut donc coordonner avec le cours porteur |
|  | 6.5 Respect des documents de conception. | -Évaluer la qualité de l’application. | -Les techniques d’assurance qualité.  -Les essais de système et/ou d’acceptation. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 7. Participer à la mise en service de l’application. | 7.1 Préparation appropriée de l’application en vue de son déploiement ou de son installation. | -Élaborer une procédure de déploiement ou d’installation  -Appliquer la procédure de déploiement ou d’installation. | -Les bonnes pratiques pour le déploiement et l’installation d’une application. |  |  |
|  | 7.2 Déploiement ou installation corrects de l’application. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DEVIS MINISTÉRIEL** | | **ORIENTATIONS LOCALES** | | | |
| **Éléments de compétence** | **Critères de performance** | **Objets d’apprentissage** | **Balises de contenu** | **Balises de l’évaluation**  **(en assurant la conformité avec la PDEA)** | **Modalités particulières**  **(au besoin)** |
| 8. Rédiger la documentation. | 8.1 Détermination correcte de l’information à rédiger. | -Utiliser les bonnes pratiques pour l’écriture de la documentation  -Écrire de la documentation pour l’usager.  -Écrire la documentation du code. | -Les bonnes pratiques pour la rédaction de la documentation.  -Les outils d’écriture de documentation.  -L’écriture de la documentation pour l’usager (ex : aide en ligne, manuel d’usager).  -L’écriture de la documentation du code (ex : commentaire de fonction, documentation d’API) | 5% |  |
|  | 8.2 Notation claire du travail effectué. |  |

|  |  |
| --- | --- |
| **Médiagraphie indicative :** |  |
|  |